BP Security Associations with Few Exchanges and EDHOC

IETF 122 LAKE WG / DTN WG

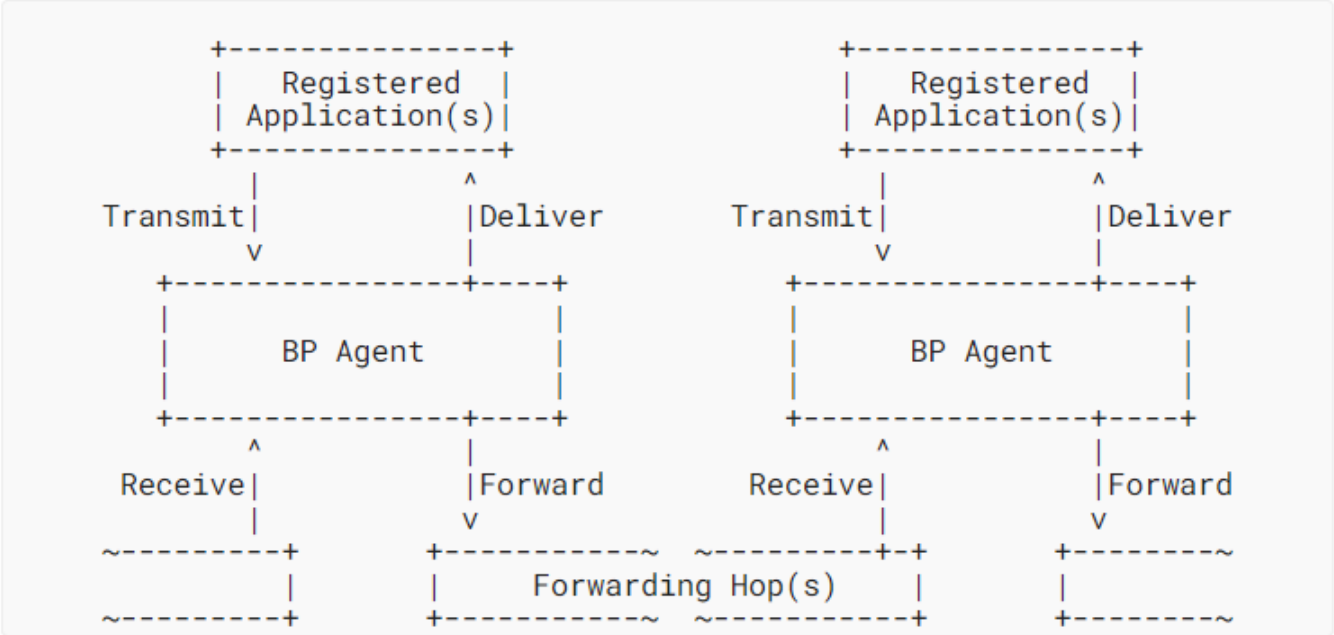
Brian Sipos
JHU/APL

Background

- The DTN Architecture is defined in RFC 4838
  - Challenged environments have high link delays, expected one-hop discontinuities, and no path guarantees
  - But these do have explicit storage and lifetime with potentially larger MTUs
- The Bundle Protocol version 7 (BPv7) is defined in RFC 9171
  - Defines a primary block “header”, extension block structure, and payload block
- Bundle Protocol Security (BPSec) is defined in RFC 9172
  - Allows fine-grained security services for integrity and confidentiality
  - Does not define when security should be applied or how to manage options and key material
  - Defines a structure for security data but not for specific algorithms
  - Delegates behavior to separate “Security Contexts” with specific parameter and result contents
- Minimum “default” security contexts for integrity and confidentiality in RFC 9173
- One draft security context is to embed COSE messages in draft-ietf-dtn-bpsec-cose
- No BPSec equivalent to IPsec IKEv2 (RFC 7296) or MACsec PBNAC (IEEE 802.1X)
  - Desire to have some inter-node autonomy to establish and maintain security associations
- EDHOC (RFC 9528) provides similar security guarantees to [D]TLSv1.3 and IKEv2
  - Done in a way which is transport-independent and extensible via external authorization data (EAD)
- The following strategy is based on the architecture described in an 2024 IEEE SMC-IT paper

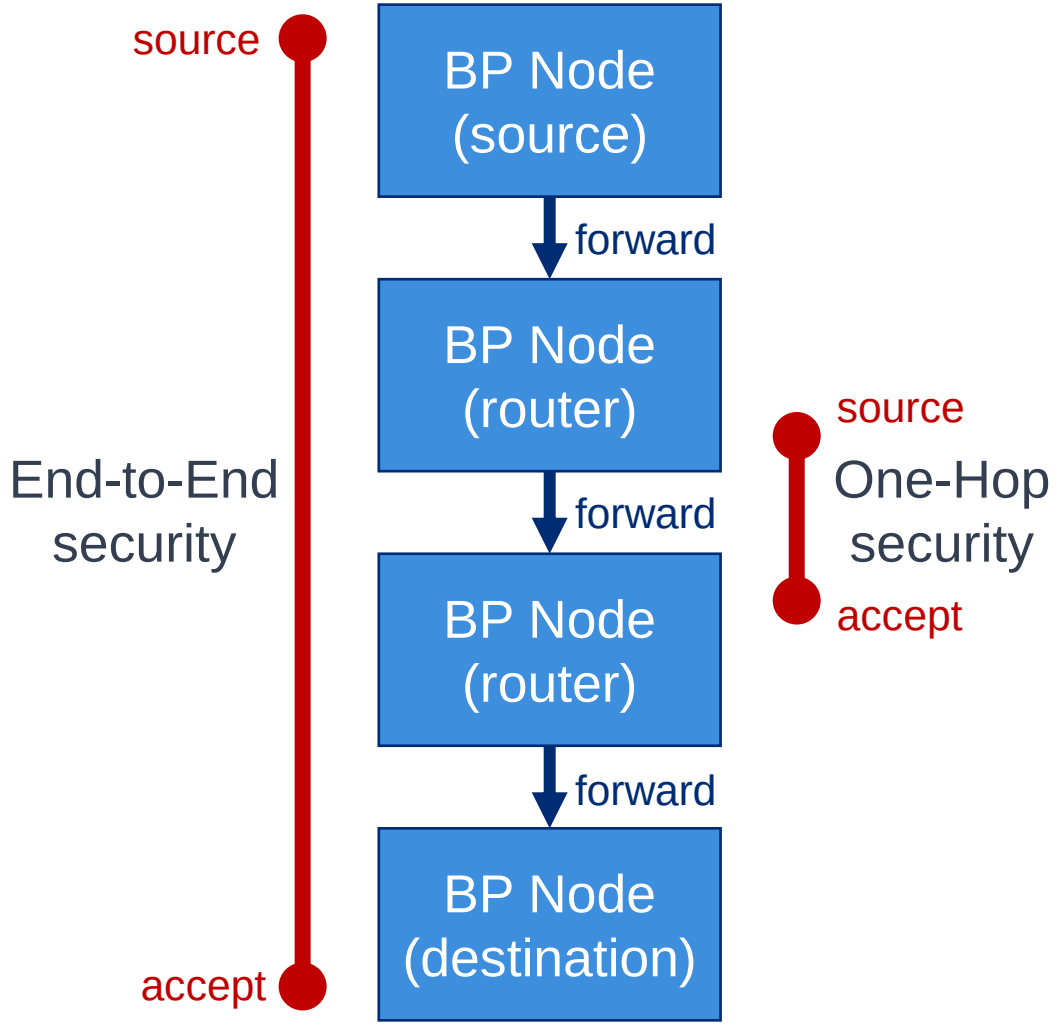
BP Agent Interaction Points and BPSec

- The formal interaction of a BP Agent with external entities in the data plane are:
  - Transmit from and deliver to registered endpoint applications
  - Receive from and forward via convergence layer adapters (CLAs)
- A BPSec entity can act as one of three roles for each operation:
  - Source – create the sec. op.
  - Verifier – verify in-place
  - Acceptor – verify and remove
- Each BP Agent can have internal long-term storage
- CLAs are defined for TCP/IP and LTP/UDP/IP among others
- Security services can be applied to bundles at each interaction point based on internal policy



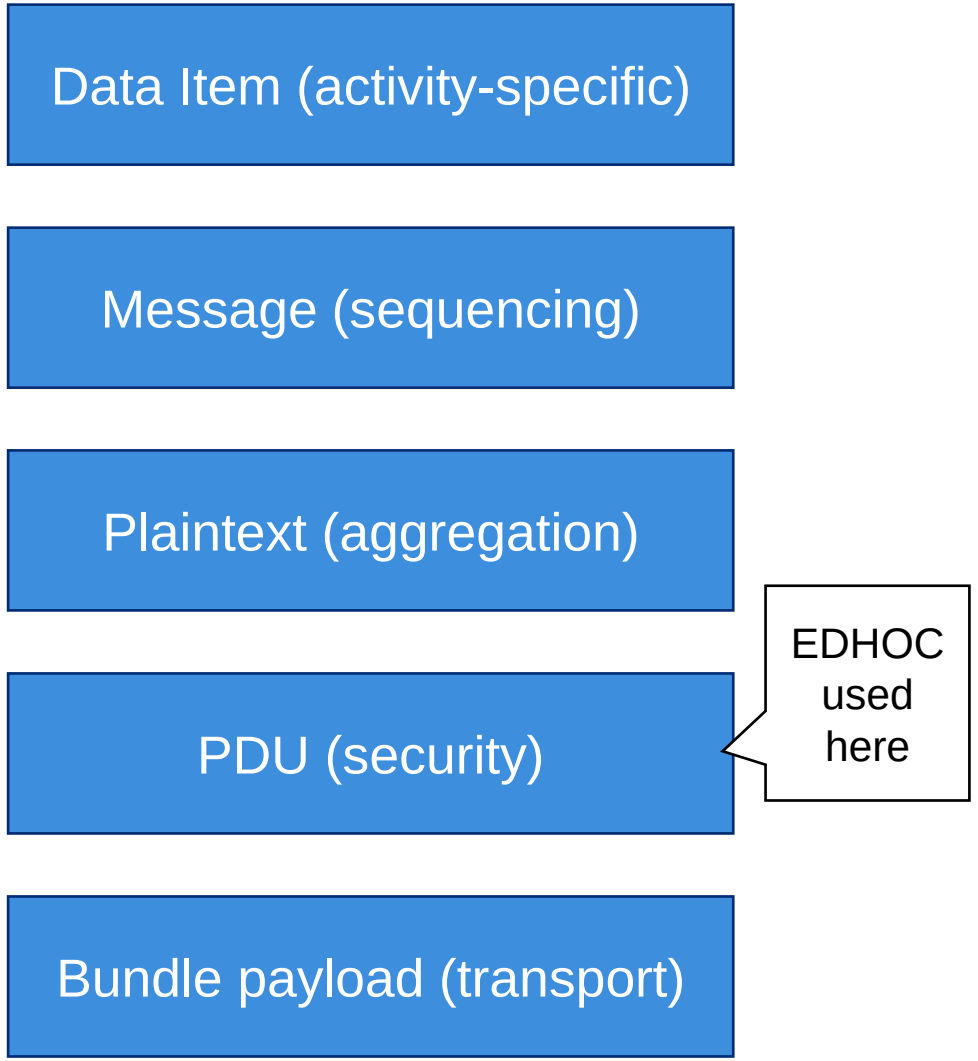
Goals of a BP Security Association Protocol

- Reuse as much infrastructure and logic as possible
  - Existing PKIX profile for authenticating BP endpoint IDs and extended key use
  - Existing strategies for SIGMA-based mutual auth'n
  - Existing two-tier SA concepts and strategies used by IKEv2 and PBNAC
  - Existing BPSec policy logic from the NASA-AMMOS BPSec Library (BSL)
- Want to operate in-band over BPv7
  - SA application must provide state and retransmission
- Need to be algorithm-agile and BPSec context-independent
  - Cannot rely on any particular BPSec context available on either node
  - Focus on roles of security source and acceptor
- Focus on near-term security use cases
  - End-to-End security between application endpoints
  - One-Hop supplement to CLA security
- Enable long-term extensibility
  - Possible to offer tunnel negotiation via (not-yet-standardized) bundle-in-bundle encapsulation (BIBE)



Messaging Strategy and Layering

- Decompose protocol behaviors into distinct “activities”
  - Each activity is a sequence of steps encoded into messages
  - The number of needed steps is activity-type-specific
  - Each message has a set of activity-type-specific data items
- Each message is encoded once
  - This is the unit of retransmission
- Allow message aggregation into plaintext
- Apply application-level security to embed plaintext into a PDU
- Transport each PDU as BP payload
  - Inform transport lifetime with retransmission timeout



EDHOC Embedding for Initial Authentication (IA)

- All of the messaging logic relies on the existence of a confidential association between peers
  - EDHOC uses ECDHE to bootstrap this
  - EDHOC EAD provides a secure channel during its session
  - EDHOC negotiates an application AEAD algorithm for post-session confidentiality
- Need to perform mutual authentication based on PKIX environment
  - EDHOC uses signature auth'n with X.509 and C509 end-entity credentials
- A two-level hierarchy of security associations requires key derivation strategies
  - EDHOC negotiates an application hash algorithm and KDF behaviors

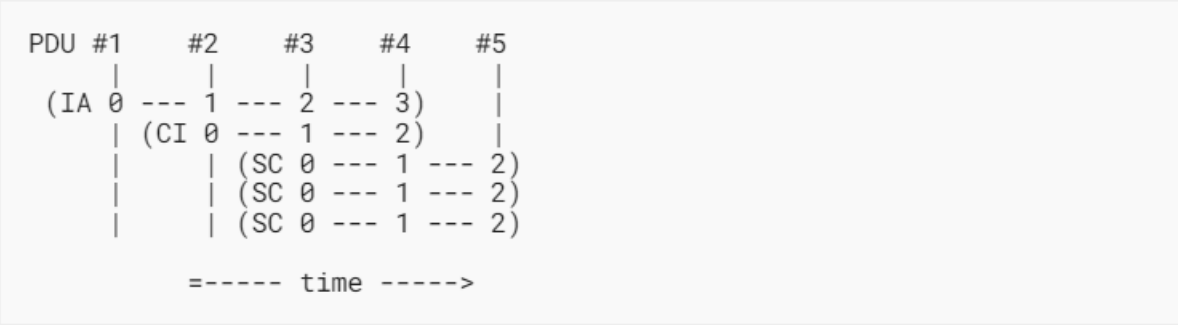


Figure 9: Pipelining with large CAS limit

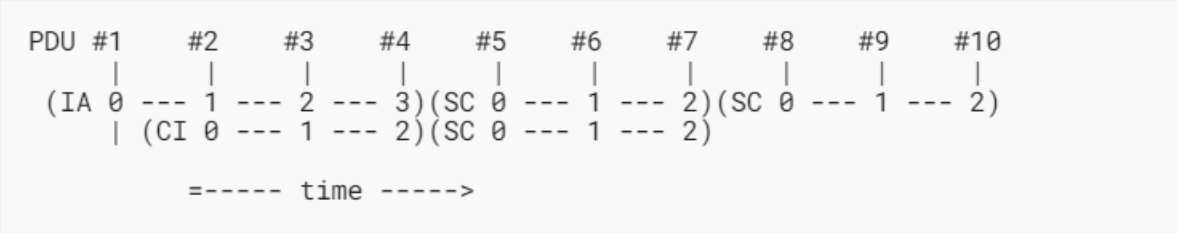


Figure 10: Pipelining with small CAS limit

Specific pipelining is based on concurrent activity support (CAS) of both entities

Prototyping Conclusions

- A proof-of-capability prototype was made in pure Python (over pycose)
- EDHOC itself was straightforward to implement, and testing was greatly simplified by using the traces from RFC 9529 as test inputs
- Directly embedding EDHOC was also straightforward, given an API focused on transport isolation, cipher suite / algorithm agility, and EAD handling
- A cipher-suite-independent unit test fixture was added to ensure self-consistency
  - This excluded the suites which rely on ChaCha20/Poly1305 which is not implemented by pycose
- The examined activity / messaging / aggregation / retransmission strategy is viable but requires detailed requirements for interoperability

Next Steps

- A protocol draft to be submitted as an IETF personal draft
  - The in-progress draft is focused on the nominal activity patterns and state paths
  - Need to address alternate patterns, failure modes, and state paths
- This protocol falls under the existing DTN WG scope for “Key Management”
  - Plan to propose BP SA protocol for WG adoption under the “Key Management Protocol” milestone
- On the EDHOC side, there is a desire to improve diagnostic tooling
  - Enable EDHOC applications to export secrets during test
  - Enable traffic diagnostic tools (*e.g.* Wireshark) to decrypt EDHOC sessions (under specific assumptions)