Securing BPSec Against Arbitrary Block Dropping

Benjamin Dowling benjamin.dowling@kcl.ac.uk
Britta Hale britta.hale@nps.edu
Xisen Tian xisen.tian1@nps.edu
Bhagya Wimalasiri b.m.wimalasiri@sheffield.ac.uk

July 23, 2025

IACR Communications in Cryptology
Vol. 0, No. 0, 26 pages.

Cryptography is Rocket Science

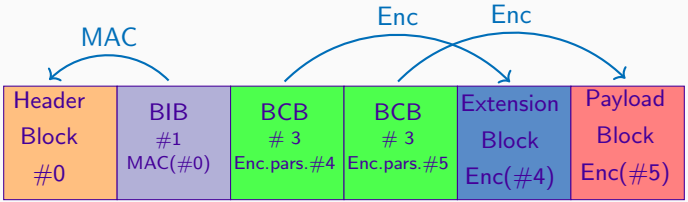
Analysis of BPSec

Benjamin Dowling[1], Britta Hale[2], Xisen Tian[2] and Bhagya Wimalasiri[3]

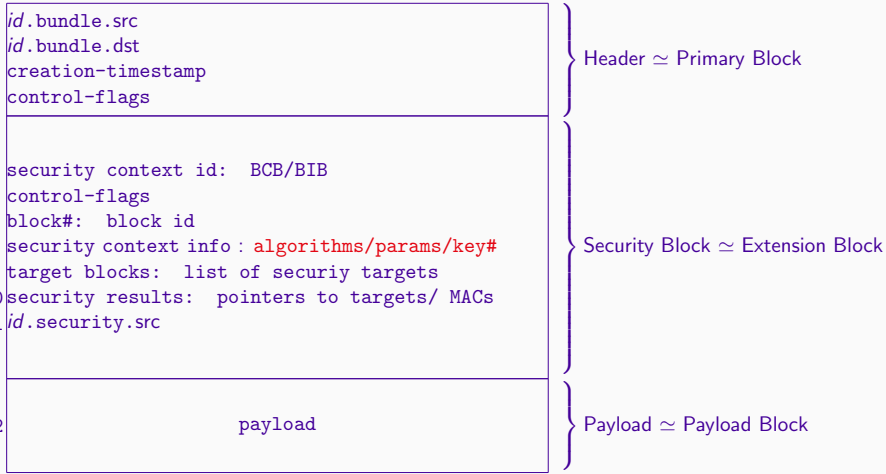
[1] King's College London, United Kingdom
[2] Naval Postgraduate School, United States
[3] University of Sheffield, United Kingdom

Bundle Protocol Security: BPSec

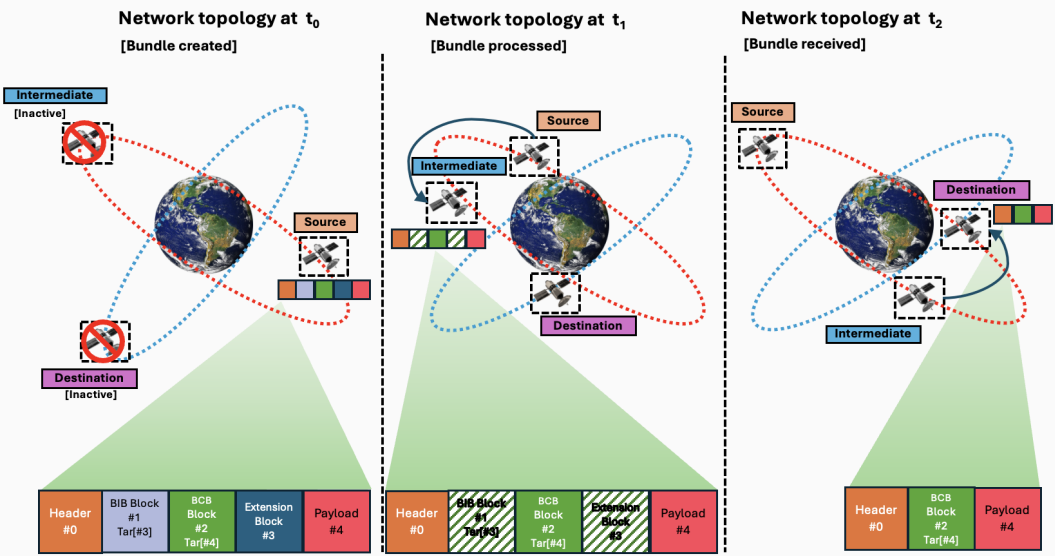
- Assumes pre-established keys/symmetric keys
- No PFS/PCS
- Security blocks are a type of extension block:
  - Bundle Confidentiality Block (**BCB**) for authenticated encryption
  - Bundle Integrity Block (**BIB**) for data authenticity
- (RFC9173) instantiates BPSec:
  - pre-shared key and key-wrapping assumptions
  - HMAC-SHA2 for BIB
  - AES-GCM for BCB



BPSec: Blocks

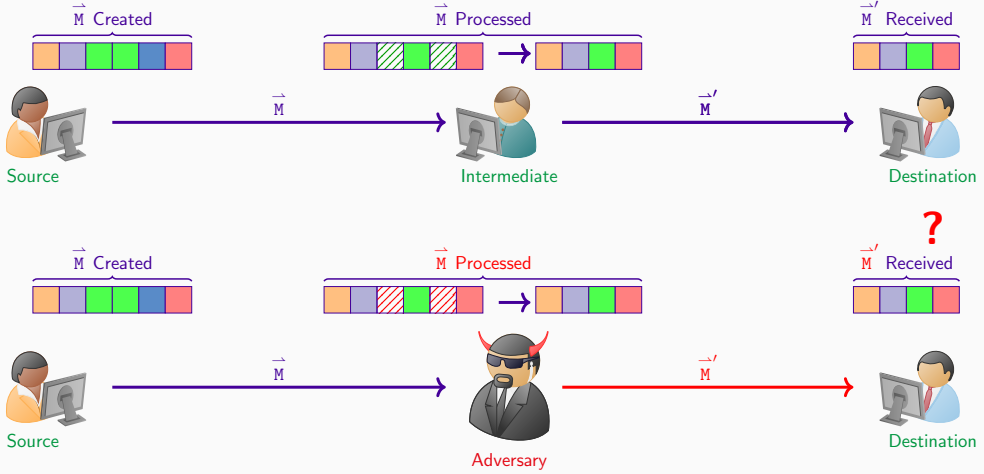


Interesting BPSec Behaviour

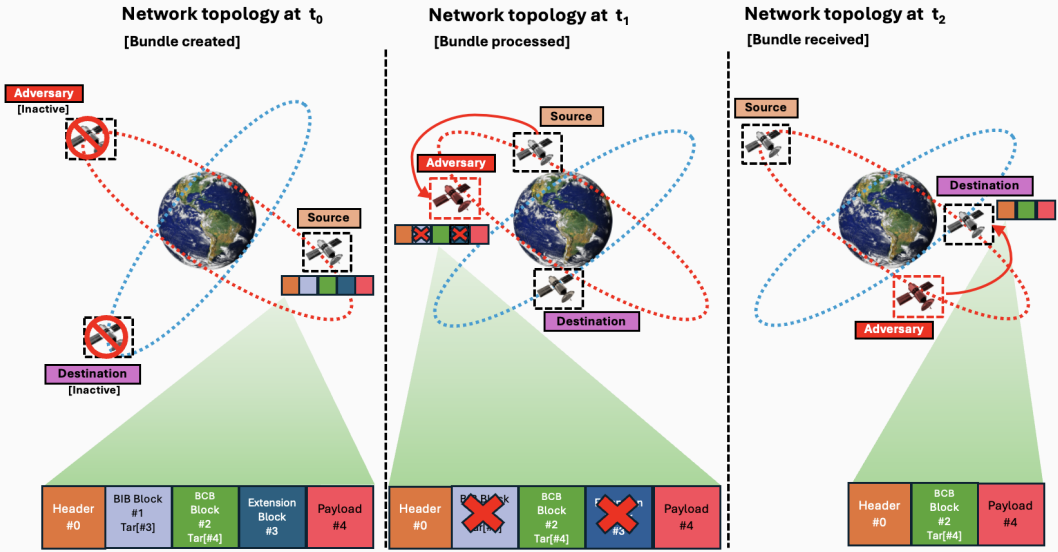


But WHO Processed the BPSec Bundle?

CANNOT differentiate between a bundle processed by an honest party or adversary



Arbitrary Message Dropping in Deepspace



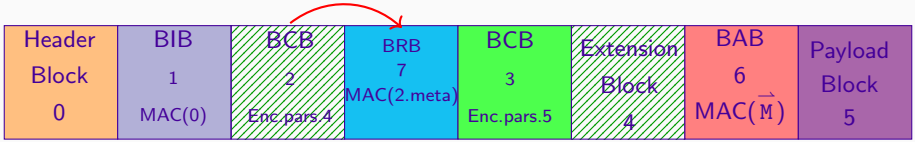
Strengthening BPSec

Add two additional layers of integrity:

Receipts (BRB) by intermediate nodes
AND
Audit MAC BAB over params and key info

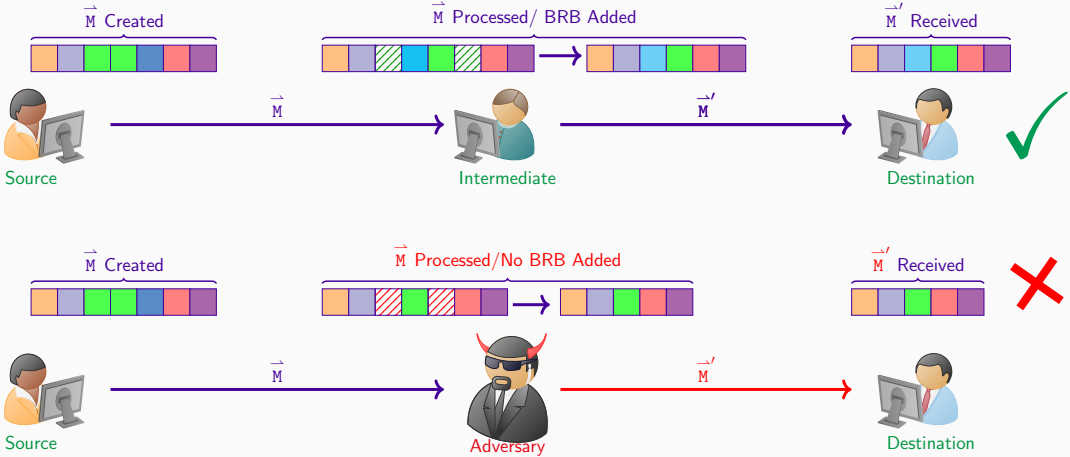
Guarantees to destination one of the following:

1. All blocks added by the source that have arrived unchanged or
2. Some honest intermediate has correctly processed and discarded block/s

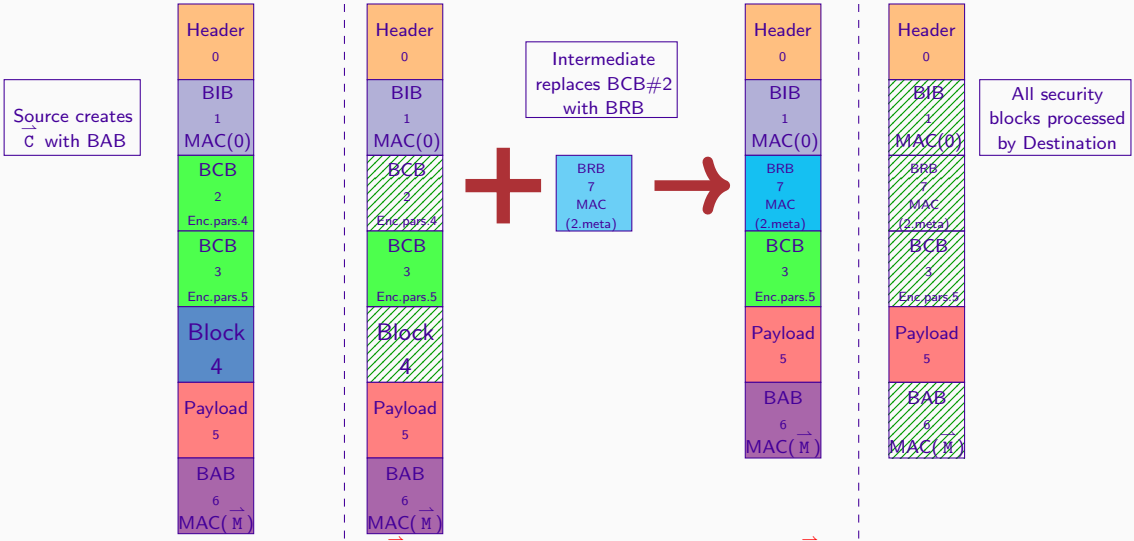


StrongBPSec **Guarantees**

Destination CAN now distinguish between a bundle processed by an honest intermediate vs. adversary



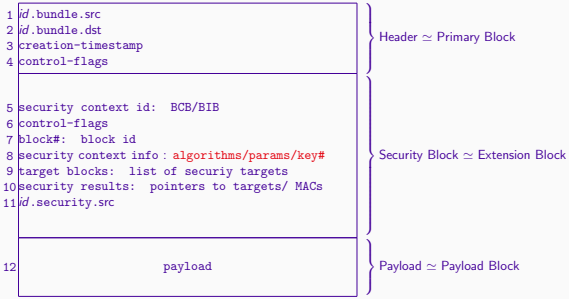
StrongBPSec **Execution**



First Draft Up

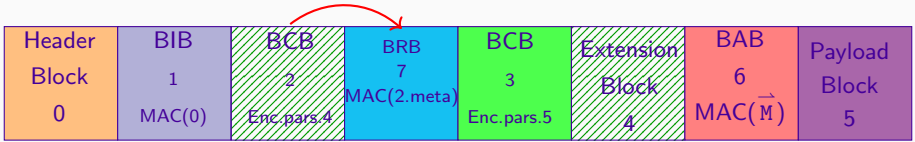
- Improved security for BPSec with StrongBPSec
- https://www.ietf.org/archive/id/draft-tian-dtn-sbam-00.txt

Request for Feedback: Unique Key Identifiers



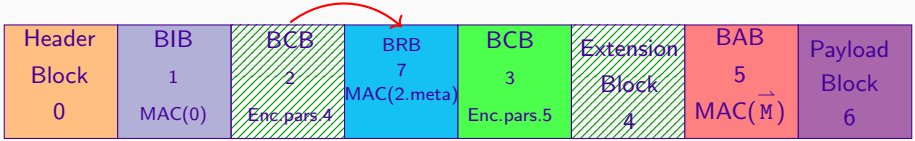
- **Assumption**: Local security policies *define* keys to use for security context.
- **Problem**: If not standardised, colliding key identifiers may impact correctness of protocol; intermediate nodes may
- **Proposal**: `key_identifier` as explicit security context parameter, enabling BPAs to uniquely identify correct keys per context.

Request for Feedback: COSE-Context Considerations



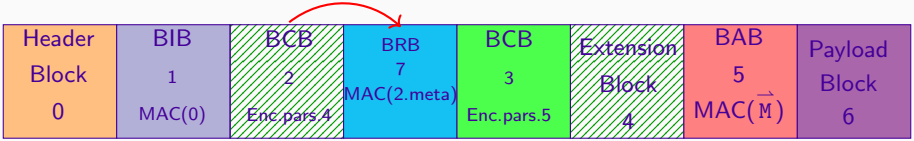
- **Setting**: SBAM, like BPSec, as specified focuses on symmetric-key primitives
- **Example**: Both BRB and BAB above are defined as MACs
- **Proposal**: Robust enough to use signature algs rather than MACs to support in the COSE-context, adds intermediate node authentication.

Request for Feedback: New Blocks BAB and BRB



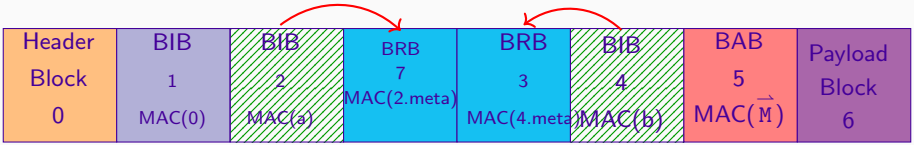
- **Assumption**: For our analysis, we assumed that BRB and BAB are additional block types, simplified analysis
- **Problem**: We now need to define new block types in the draft
- **Feedback**: Is this fine? We swapped ordering (BAB $<-->$ Payload) due to feedback.

Request for Feedback: Blocks Excluded by SBAM



- **Problem:** Blocks Excluded by SBAM. Need to identify which ones.
  - Blocks that necessarily change over the course of a bundle lifespan (e.g. previous hop)
  - Bundle-In-Bundle-Encapsulation blocks?
  - Others?

Request for Feedback: Possible Efficiency Extension?



- **Problem:** SBAM introduces overhead: Need to replace BIB/BCB with BRB.
- **Proposal:** Collapsing multiple BRBs to save total number of blocks sent (e.g. if multiple BRBs are referring to a set of common operations performed by an intermediate processor) similar to how one BIB can have multiple targets?
  - This has not yet been analysed or fully developed!

Summary

- Improved security for BPSec with StrongBPSec
- https://www.ietf.org/archive/id/draft-tian-dtn-sbam-00.txt