ION Open Source 4.1.2 BPSec Release Notes

Overview of BPSec in 4.1.2

The implementation of BPSec in ION Open Source (IOS) 4.1.2 is not feature-complete and is considered to be a “beta” version of BPSec. This early release of the code allows users to begin to integrate BPSec into their projects, as core functionality is implemented, as well as encourages early testing and feedback for the major BPSec release in IOS 4.2.

A feature-complete implementation of BPSec (RFC 9172) will be provided in the ION Open Source 4.2 release.

Implemented Features of BPSec

1. Security Services:
    a. bib-integrity.
    b. bcb-confidentiality.
2. Security roles – Security Source, Verifier, and Acceptor, for both security services with the exception of the Security Verifier role for bcbc-confidentiality.
    a. A Security Source adds a security operation to a bundle.
    b. A Security Verifier checks a security operation in the bundle.
    c. A Security Acceptor processes a security operation (verifies integrity/decrypts the target block) in the bundle.
3. Supported security targets: all blocks, including the Primary Block, Payload Block, and extension blocks for both bib-integrity and bcb-confidentiality security services, except where prohibited by RFC 9172.
    a. Primary Block: bib-integrity
    b. Payload Block: bib-integrity, bcb-confidentiality
    c. Non-Security Extension Blocks: bib-integrity, bcb-confidentiality
4. Security Contexts:
    a. ION Test Security Context (ITSC): The ITSC can be used for both bib-integrity and bcb-confidentiality security operations and supports the BPSec capabilities provided with previous ION releases for backward compaitibility:
        i. bib-integrity operations on the Primary Block.
        ii. bib-integrity operations on the Payload Block.
        iii. bcb-confidentiality operations on the Payload Block.
    b. BIB-HMAC-SHA (RFC 9173)
    c. BCB-AES-GCM (RFC 9173)
5. Security Policy, to include:
    a. Removal of deprecated transmit/receive BIB and BCB rules.
    b. BPSec policy rules, configured using bpsecadmin.
    c. Security operation events and event sets.

   d. Optional processing actions: remove security operation, remove security target, and remove all security target operations.
   e. For an in-depth description of BPSec policy changes and features, please consult the user's manual at /bpv7/bpsec/policy/Security_Policy_User_Manual.docx.

BPSec Features Not Implemented

1. Target multiplicity. Security blocks are currently limited to a single security target block.
2. Advanced Security Block interactions.
   a. Multiple security services applied to a single security target. Ex: a shared security target between a BIB and BCB.
   b. BIB encryption.
3. The Security Verifier role for bcb-confidentiality.
4. Enforcement of required processing actions for security operation events is incomplete. For example, the security operation is not always removed from the bundle when it is identified as corrupted at its security acceptor.
5. Identification of missing security operations at a Security Verifier and Acceptor.
6. Identification of unexpected security operations – that is, security operations that should not be represented in the bundle but are present.
7. Bundle destination BPSec behavior – a bundle destination is required to be the security acceptor for any remaining security operations in the received bundle.
8. Anonymous bundle handling.
9. Key wrap has not been tested and will be completed for the 4.2 release.
10. The following optional security policy actions are not yet supported:
    a. Request storage of a bundle that should not be forwarded.
    b. Override target block processing control flags
    c. Override security block processing control flags.

Testing

The BPSec implementation has been tested for the nominal path, including handling the Security Source, Verifier, and Acceptor roles for bib-integrity and Security Source and Acceptor roles for bcb-confidentiality. The BPSec python test framework, provided in /tests/bpsec/python_tests, was used to conduct this testing and resulted in all test cases passing.

For details on individual test cases and their results, please consult /tests/bpsec/python_tests/BPSec_Python_Test_Suite.docx. Note that none of these tests cover the generation of cryptographic material as this is not supported by ION Open Source releases.

Known Errors

The following are known or potential issues identified for this release (4.1.2). Corrections will be made in the IOS 4.2 release.

- Optional security processing actions ‘do not forward’ and ‘report reason code’ will be updated for the IOS 4.2 release and have not been tested for this release.
- Execution of multiple security processing actions in response to the occurrence of a single security operation event has not been tested for this release.
- Providing malformed security policy input to bpsecadmin may cause the utility to be unable to execute the “quit” command, requiring the user to kill the process instead.
- Limitations to SDR nested transactions in ION can cause the scenario where two SDR objects in the same work flow are handled in a single nested transaction. Because this single transaction provides only one “modified” flag, it cannot be set properly to track both SDR objects. This limitation impacts BPSec block processing and is being worked as issue #61 “ici transaction "modified" flag bug” on the OU Gitlab.

Logging

By default, the BPSec logging level is set to “error”, reporting only critical security errors in the ion.log file. The logging level (error, warning, information, and process)can be changed to provide additional detail.

There are four levels of debugging specified for BPSec:

1. Function entry/exit logging. This logs the entry and exit of all major functions in the BPSec library and is useful for confirming control flow through the BIB/BCB modules.
2. Information logging. Information statements are peppered through the code to provide insight into the state of the module at processing points considered useful by bpsec module software engineers.
3. Warning logging. Warning statements are used to flag unexpected values that, based on context, may not constitute errors.
4. Error logging. Errors are areas in the code where some sanity check or other required condition fails to be met by the software.

Debugging can be turned off at compile time by removing the BIB_DEBUGGING. BCB_DEBUGGING, and/or BPSEC_DEBUGGING #defines.

Note: Test logging for both BIB and BCB is disabled by default. To enable test-level logging statements, enable BIB_TEST_LOGGING and BCB_TEST_LOGGING in bib.h and bcb.h, respectively. Details can be found in Section 2, Configuring ION for BPSec Testing, in the test manual: /tests/bpsec/python_tests/BPSec_Python_Test_Suite.docx.

Contact

BPSec was implemented by The Johns Hopkins University Applied Physics Laboratory for the IOS 4.1.2 release. Feedback on the existing implementation and improvements for the 4.2 release is welcome.

Ed Birrane
Edward.Birrane@jhuapl.edu

Sarah Heiner
Sarah.Heiner@jhuapl.edu