JOHNS HOPKINS
APPLIED PHYSICS LABORATORY

11100 Johns Hopkins Road
Laurel, MD 20723-6099

BPSec Security Policy Architecture

IETF 111

Sarah E. Heiner
Johns Hopkins University Applied Physics Laboratory
Sarah.Heiner@jhuapl.edu

Introduction

- The next step for BPSec is the development of security policy
  - Compliment the features of BPSec
  - Provide configuration options for mission adoption
  - Create a flexible, user-friendly framework

- Discuss current, proposed security policy architecture and the associated implementation

Agenda

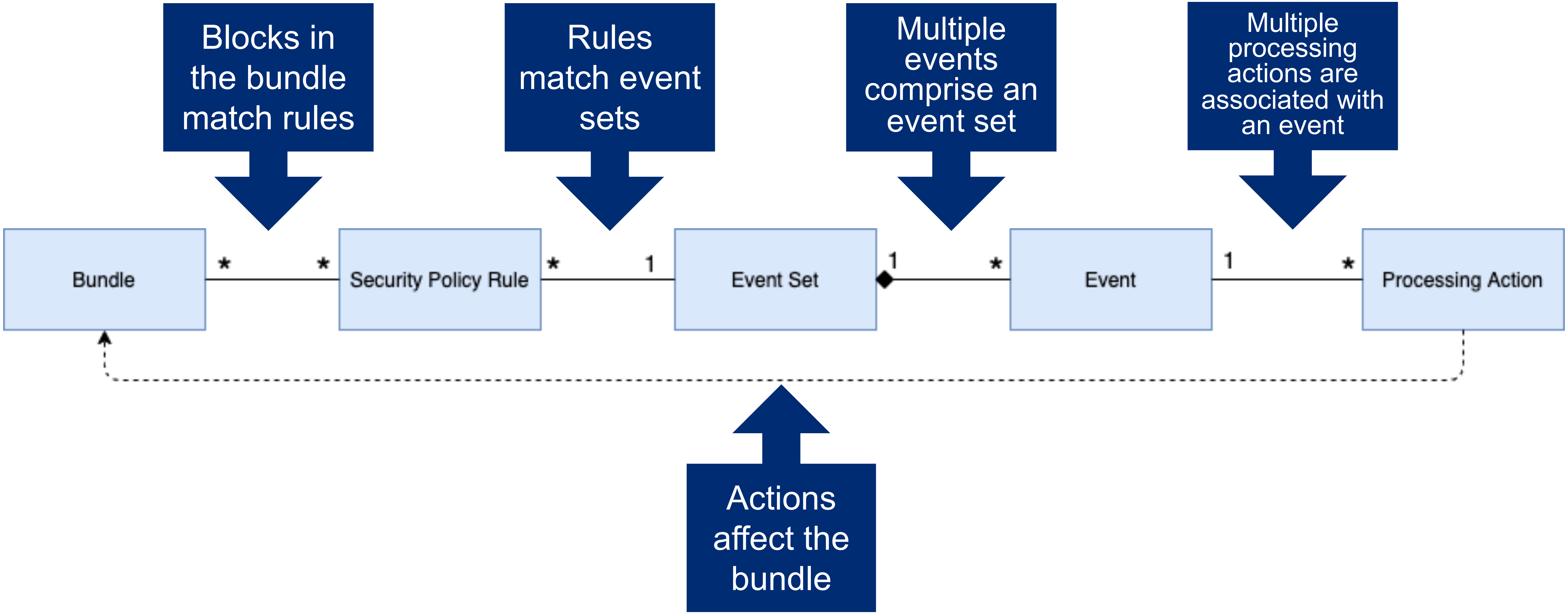
- Proposed design principles for BPSec policy
- A data model for security policy
- Security policy rules
- Security events and the actions associated with each
- Restrictions for policy actions

BPSec Policy Design

Property	Rationale
Syntactic Interoperability	Policy must result in bundle and blocks that are parsable by all security-processing nodes in the network.
Semantic Interoperability	Policy must result in a deterministic, coherent behavior within the network.
Efficient Processing	Policy must be enforceable within the likely resource constraints of spacecraft
Block Granularity	Policy must have the same maximum resolution as the BPSec allows.
Node Customizability	Policy must fit the capabilities of the node on which it is deployed.

The BPSec policy framework must be flexible and featureful

The Security Policy Data Model



Blocks in the bundle match rules

Rules match event sets

Multiple events comprise an event set

Multiple processing actions are associated with an event

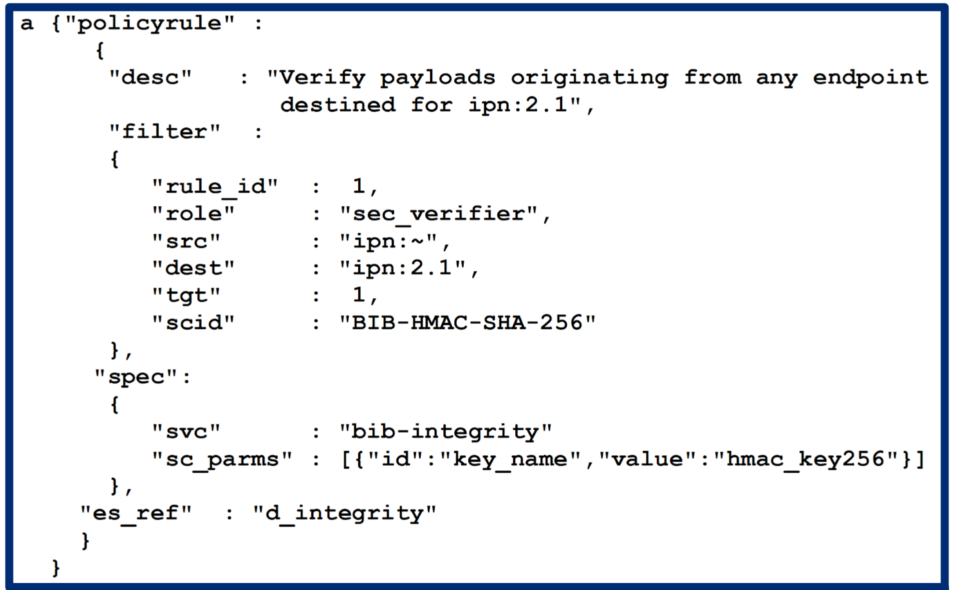
Actions affect the bundle

Security Policy Rules

- **Filter Criteria**
  - The bundle(s) the rule applies to
  - The block(s) in those bundles that are security targets of the specified security operation
  - The security policy role the BPA applying the rule must play

- **Specification Criteria**
  - Security service
  - Security context

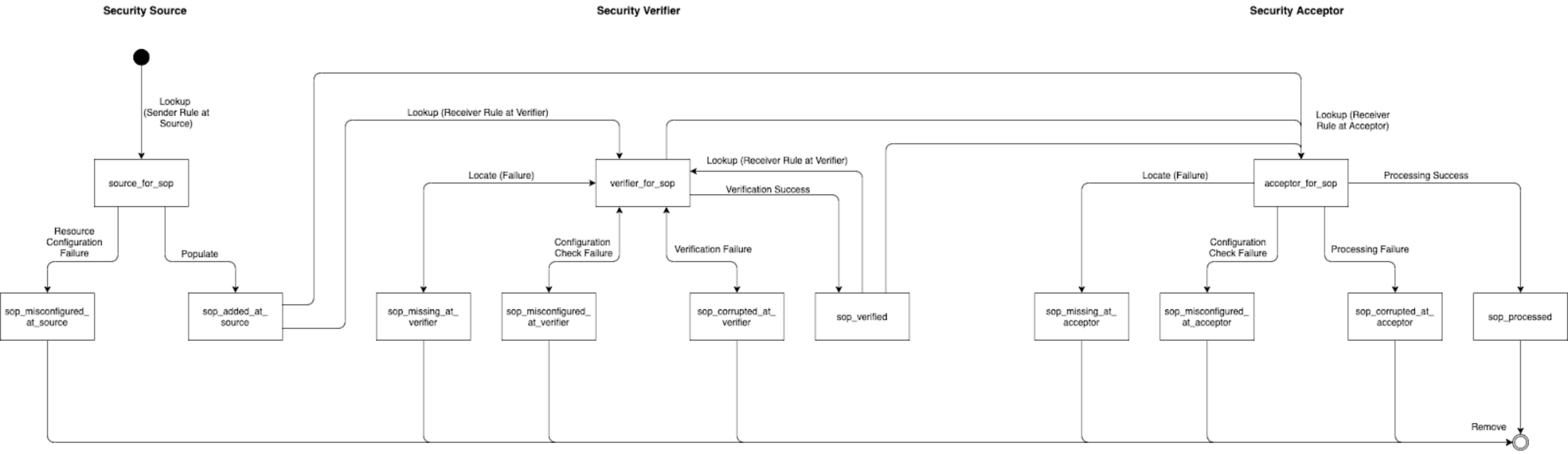
- **Event Criteria**
  - Association with an event set



```
a {"policyrule" :
     {
      "desc"    : "Verify payloads originating from any endpoint
                   destined for ipn:2.1",
      "filter"  :
      {
         "rule_id"  :  1,
         "role"     : "sec_verifier",
         "src"      : "ipn:~",
         "dest"     : "ipn:2.1",
         "tgt"      :  1,
         "scid"     : "BIB-HMAC-SHA-256"
      },
     "spec":
      {
         "svc"      : "bib-integrity"
         "sc_parms" : [{"id":"key_name","value":"hmac_key256"}]
      },
    "es_ref"  : "d_integrity"
    }
  }
```

Sample Security Policy Rule

The Security Operation Lifecycle


Security Source
Security Verifier
Security Acceptor
Lookup (Sender Rule at Source)
source_for_sop
Resource Configuration Failure
Populate
sop_misconfigured_at_source
sop_added_at_source
Lookup (Receiver Rule at Verifier)
Locate (Failure)
verifier_for_sop
Lookup (Receiver Rule at Verifier)
Verification Success
Configuration Check Failure
Verification Failure
sop_missing_at_verifier
sop_misconfigured_at_verifier
sop_corrupted_at_verifier
sop_verified
Lookup (Receiver Rule at Acceptor)
Locate (Failure)
acceptor_for_sop
Processing Success
Configuration Check Failure
Processing Failure
sop_missing_at_acceptor
sop_misconfigured_at_acceptor
sop_corrupted_at_acceptor
sop_processed
Remove

Security events are the processing points for the application of security policy

Security Operation Events

- Are security failures captured sufficiently?
  - ○ Missing
  - ○ Misconfigured
  - ○ Corrupted

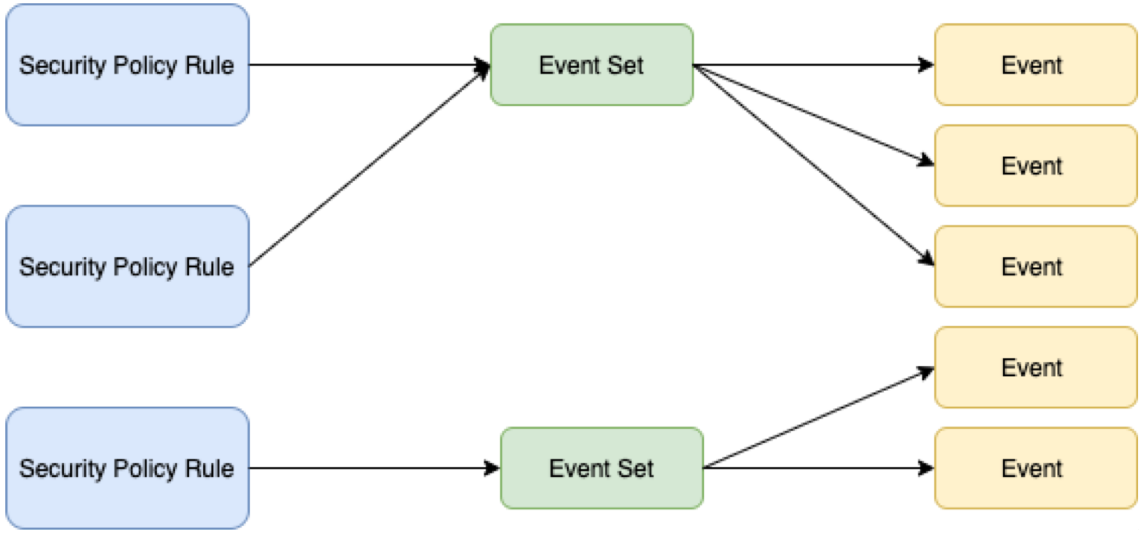
- Are there other events in the successful path that may be encountered?

```
1.  "source_for_sop”
2.  "sop_added_at_source”
3.  "sop_misconfigured_at_source”
4.  "verifier_for_sop”
5.  "sop_misconfigured_at_verifier”
6.  "sop_missing_at_verifier”
7.  "sop_corrupted_at_verifier”
8.  "sop_verified”
9.  "acceptor_for_sop”
10."sop_misconfigured_at_acceptor”
11."sop_missing_at_acceptor”
12."sop_corrupted_at_acceptor”
13."sop_processed"
```

Security Event Sets

- Set of security events associated with processing actions
  - Named
  - Re-useable

- Support generalized responses to security events



Security event sets support default security policy configurations

Processing Actions

- Retain Security Operation
- Remove Security Operation
- Remove Security Operation Target
- Remove All Security Target Operations
- Fail Bundle Forwarding
- Request Bundle Storage
- Report Reason Code
- Override Security Target's Block Processing Control Flags
- Override Security Block's Block Processing Control Flags

Categories:

- Block Manipulation
- Bundle Manipulation
- Data Generation

Processing actions are

- **Required**
- **Optional**
- **Prohibited**

for security events

Mapping: Processing Actions to Lifecycle Events

	A1	A2	A3	A4	A5	A6	A7	A8	A9
E1									
E2		O	O	O	O	O	O		
E3	R								O
E4	R								
E5		O	O		O	O	O	O	O
E6			O		O	O	O	O	
E7		O	O	O	O	O	O	O	O
E8	R								
E9									
E10		R	O		O	O	O	O	O
E11			O		O	O	O	O	
E12		R	O	O	O	O	O	O	O
E13		R							

Bundle Manipulation Processing Actions

- Retain Security Operation
- Remove Security Operation
- Remove Security Operation Target
- Remove All Security Target Operations
- Fail Bundle Forwarding

Application of these processing actions affects the bundle being processed by:

- **Modifying bundle transmission**
- **Modifying bundle contents**

Block Manipulation Processing Actions

- Override the **security target block's** block processing control flags
- Override the **security operation's** block processing control flags
- Impacts:
  - Block replication
  - Status reporting
  - Bundle/block preservation

Application of these processing actions affects a block in the bundle by:

- **Temporarily Overriding**
- **Modifying**

block processing control flags

Data Generation Processing Actions

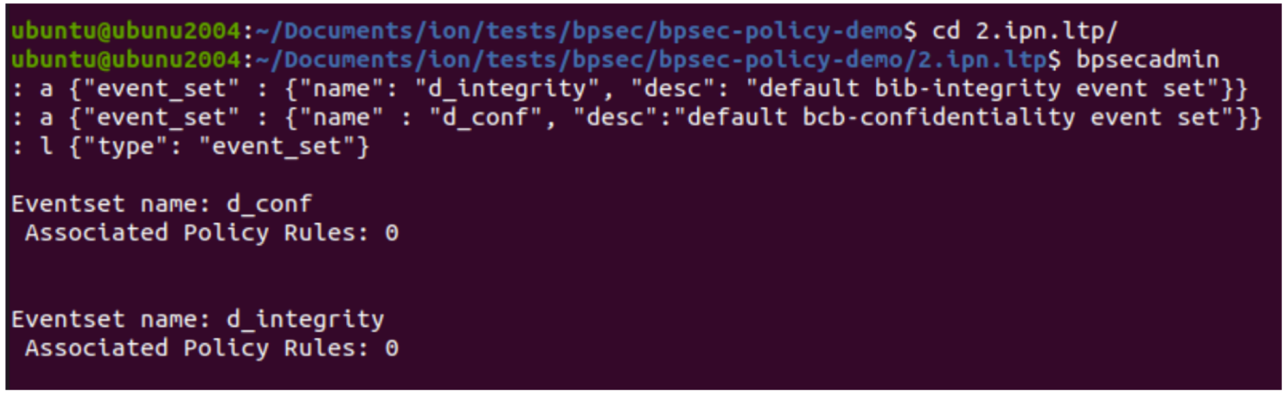
- Report occurrence of the security operation event with reason code
- Request storage of the bundle at the current node

Application of these processing actions creates data to be used for later forensic analysis by:

- **Creating a bundle status report**
- **Storing the bundle as-is**

Initial BPSec Policy Implementation in ION

- Built on the Bundle Protocol version 7 and BPSec implementations in ION
- Security policy is configured using the bpsecadmin utility
- Use of JSON and jsmn parser
  - ○ Expressive, structured syntax
  - ○ Ability to capture the possibilities of configuration while remaining consistent
- Available in ION 4.0.2 and later versions



```
ubuntu@ubunu2004:~/Documents/ion/tests/bpsec/bpsec-policy-demo$ cd 2.ipn.ltp/
ubuntu@ubunu2004:~/Documents/ion/tests/bpsec/bpsec-policy-demo/2.ipn.ltp$ bpsecadmin
: a {"event_set" : {"name": "d_integrity", "desc": "default bib-integrity event set"}}
: a {"event_set" : {"name" : "d_conf", "desc":"default bcb-confidentiality event set"}}
: l {"type": "event_set"}

Eventset name: d_conf
 Associated Policy Rules: 0


Eventset name: d_integrity
 Associated Policy Rules: 0
```

Security policy must be both expressive and consistent

Additional Information

- Security policy initial implementation in ION 4.0.2 and later
- ION Demo: Security Policy
  - https://www.youtube.com/watch?v=RW-MQuJYoG0
- Security Policy User’s Manual
- Engineering materials: Requirements and Design documentation for security policy
- SMC-IT STINT Talk: BPSec Policy in ION
- SCC Paper: Towards an Interoperable Security Policy for Space-Based Internetworks

APL

JOHNS HOPKINS

APPLIED PHYSICS LABORATORY