Delay-Tolerant Networking
Internet-Draft
Intended status: Informational
Expires: 28 April 2022
October 2021

E.J. Birrane
E. Annis
S.E. Heiner
Johns Hopkins Applied Physics Laboratory

Asynchronous Management Architecture
draft-ietf-dtn-ama-03

Abstract

This document describes a management architecture suitable for deployment in challenged networking environments for the configuration, monitoring, and local control of application services. Challenged networking environments exhibit interruptions in end-to-end connectivity and communications delays that are both long-lived and unpredictable. Even in these challenging conditions, such networks must provide some type of end-to-end information transport and fault protection while also supporting configuration and performance reporting. This management may need to operate without human- or system-in-the-loop synchronous interactivity and without the preservation of transport-layer sessions. In such a context, challenged networks must exhibit behavior that is both determinable and autonomous while maintaining as much compatibility with non-challenged-network operational concepts as possible.

The architecture described in this document is termed the Asynchronous Management Architecture (AMA). The AMA supported two types of asynchronous behavior. First, the AMA does not presuppose any synchronized transport behavior between managed and managing devices. Second, the AMA does not support any query-response semantics. In this way, the AMA allows for operation in extremely challenging conditions, to include over uni-directional links and cases where delays/disruptions would otherwise prevent operation over traditional transport layers, such as when exceeding the Maximum Segment Lifetime (MSL) of the Transmission Control Protocol (TCP).

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Asynchronous Management Architecture (AMA) provides a novel approach for the configuration, monitoring, and local control of application services on a managed device over a challenged network. The unique properties of a challenged network are as defined in [RFC7228] and include cases where an end-to-end transport path may not be feasible at any moment in time and delivery delays may prevent timely communications between a network operator and a managed device. These delays may be caused by long signal propagations or frequent link disruptions (such as described in [RFC4838]) or by non-environmental factors such as quality-of-service prioritizations and service-level agreements.

Importantly, the management approach for a challenged network must be one which remains operational in the most restrictive environments in which such networks might be instantiated. The AMA approach should be functional in a variety of potential management scenarios, to include the following.

* Managed devices that are only accessible via a uni-directional link, or via a link whose duration is shorter than a single round-trip propagation time.

* Links that may be significantly constrained by capacity or reliability, but at (predictable or unpredictable) times may offer significant throughput.

* Multi-hop challenged networks that interconnect two or more unchallenged networks such that managed and managing devices exist in different networks.

In these and related scenarios, managed devices need to operate with a certain level of local autonomy because managing devices may not be available within operationally-relevant timeframes. Managing devices deliver instruction sets that govern the local, autonomous behavior of the managed device. These behaviors include, but are not limited to, collecting performance data, state, and error conditions, and applying pre-determined responses to pre-determined events.

The AMA is a novel approach to management that can leverage transport, network, and security solutions designed for challenged networks, but is not bound to any single solution. The goal is asynchronous communication between the device being managed and the manager, at times never expecting a reply, and with knowledge that commands and queries may be delivered much later than the initial request.

More generally, the AMA approach is designed such that it can be deployed in all environments in which the Delay/Disruption-Tolerant (DTN) Bundle Protocol (BPv7) [I-D.ietf-dtn-bpbis] may be deployed.

1.1. Scope

This document describes the motivation, services, desirable properties, roles/responsibilities, logical data model, and system model that form the AMA. These descriptions comprise a concept of operations for management in challenged networks with sufficient specificity that implementations conformant with this architecture will operate successfully in a challenged networking environment.

The AMA described herein is strictly a framework for application management over a challenged network. The document is not a prescriptive standardization of a physical data model or any protocol. Instead, it serves as informative guidance to authors and users of such models and protocols.

The AMA is independent of transport and network layers. It does not, for example, require the use of TCP or UDP. Similarly, the AMA does not pre-suppose the use of IPv4 or IPv6.

The AMA is not bound to a particular security solution. It is assumed that any network using this architecture supports those services such as naming, addressing, integrity, confidentiality, and authentication required to communicate AMA messages. Therefore, the transport of these messages is outside of the scope of the AMA.

While possible that a challenged network may interface with an unchallenged network, this document does not address the concept of compatibility with other management approaches.

1.2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

1.3. Organization

The remainder of this document is organized into seven sections that, together, describe an AMA suitable for management of challenged networks. The description of each section is as follows.

* Terminology - This section identifies those terms critical to understanding the proper operation of the AMA. Whenever possible, these terms align in both word selection and meaning with their analogs from other management protocols.

* Motivation - This section provides an overall motivation for this work as providing a novel and useful alternative to other network management approaches.

* Services - This section identifies and defines the services that an AMA will provide to network and mission operators that are unique to operating in a challenged environment.

* Desirable Properties - This section identifies those properties of a challenged network management system required to effectively implement needed services. These properties guide the subsequent definition of the system and logical models that comprise the AMA.

* Roles and Responsibilities - This section identifies roles in the AMA and their associated responsibilities. It provides the context for discussing how services are provided by both managers and agents.

* Logical Data Model - This section describes the kinds of data, procedures, autonomy, and associated hierarchal structure inherent to the AMA.

* System Model - This section describes data flows amongst various defined AMA roles. These flows capture how the AMA system works to manage devices across a challenged network.

2. Terminology

* Actor - A software service running on either managed or managing devices for the purpose of implementing management protocols between such devices. Actors may implement the "Manager" role, "Agent" role, or both.

* Agent Role (or Agent) - A role associated with a managed device, responsible for reporting performance data, accepting/performing controls, error handling and validation, and executing any autonomous behaviors. AMA Agents exchange information with AMA Managers operating either on the same device or on a remote managing device.

* Asynchronous Management - Management that does not depend on stateful connections or real time delivery of management messages. Allows for delivery of management messages and instruction sets for autonomous behavior that governs the expected actions, rules associated with those actions, and expected reporting procedures. Asynchronous management does not depend on underlying transport or network protocols for reliability or addressing of source and destination.

* Asynchronous Management Model (AMM) - data types and data structures needed to manage applications in asynchronous networks.

* Externally Defined Data (EDD) - Information made available to an AMA Agent by a managed device, but not computed directly by the AMA Agent itself.

* Variables (VARs) - Typed information that is computed by an AMA Agent, typically as a function of EDD values and/or other variables.

* Constants (CONST) - A constant represents a typed, immutable value that is referred to by a semantic name. Constants are used in situations where substituting a name for a fixed value provides useful semantic information. For example, using the named constant PI rather than the literal value 3.14.

* Controls (CTRLs) - Procedures run by an AMA Actor to change the behavior, configuration, or state of an application or protocol being asynchronously managed. Controls may also be used to request data from an agent and define the rules associated with generation and delivery.

* Literals (LITs) - A literal represents a typed value without a semantic name. Literals are used in cases where adding a semantic name to a fixed value provides no useful semantic information. For example, the number 4 is a literal value.

* Macros (MACROs) - A named, ordered collection of Controls and/or other Macros.

* Manager Role (or Manager) - A role associated with a managing device responsible for configuring the behavior of, and eventually receiving information from, AMA Agents. AMA Managers interact with one or more AMA Agents located on the same device and/or on remote devices in the network.

* Operator (OP) - The enumeration and specification of a mathematical function used to calculate variable values and construct expressions to evaluate AMA Agent state.

* Report (RPT) - A typed, ordered collection of data values gathered by one or more AMA Agents and provided to one or more AMA Managers. Reports only contain typed data values and the identity of the Report Template (RPTT) to which they conform.

* Report Template (RPTT) - A named, typed, ordered collection of data types that represent the structure of a report (RPT). This is the schema for a report, generated by an AMA Manager and communicated to one or more AMA Agents.

* Rule - A unit of autonomous specification that provides a stimulus-response relationship between time or state on an AMA Agent and the actions or operations to be run as a result of that time or state. A rule might trigger updating a variable, populating a report/table, executing a control, or initiating the transmission of a report/table.

* State-Based Rule (SBR) - A state-based rule is any rule in which the rule stimulus is triggered by the calculable internal state of data model associated with the AMA Agent.

* Synchronous Management - Management that assumes messages will be delivered and acted upon in real or near-real-time. Synchronous management often involves immediate replies of acknowledgment or

error status. Synchronous management is often bound to underlying transport protocols and network protocols to ensure reliability or source and sender identification.

* Table (TBL) - A typed collection of data values organized in a tabular way in which columns represent homogeneous types of data and rows represent unique sets of data values conforming to column types. Tables only contain typed data values and the identity of the Table Template (TBLT) to which they conform.

* Table Template (TBLT) - A named, typed, ordered collection of columns that comprise the structure for representing tabular data values. This template forms the structure of a table (TBL).

* Time-Based Rule (TBR) - A time-based rule is a specialization, and simplification, of a state-based rule in which the rule stimulus is triggered by the relative time as it is known on the Agent as a function of either matched value or frequency.

3. Motivation

Early work into the rationale and motivation for specialized management for challenged networks was captured in [BIRRANE1], [BIRRANE2], and [BIRRANE3]. Some of the properties and feasibility of such a management system were adopted from prototyping work done in accordance with the DTN Research Group within the IRTF as documented in [I-D.irtf-dtnrg-dtnmp].

The unique nature of challenged networks requires new network capabilities to deliver expected network functions. For example, the unique nature of DTNs required the development of the Bundle Protocol for transport functions and the Bundle Protocol Security Protocol (BPSec) is required to secure bundles in certain types of DTNs. Similarly, new management capabilities are needed to implement management in challenged environments, such as those defined as DTNs.

The AMA provides a method of configuring AMA Agents with local, autonomous management functions, such as rules-based execution of procedures and generation of reports, to achieve expected behavior when managed devices exist over a challenged network. It further allows for dynamic instantiation and population of Variables and reports through local operations defined by the manager, as well as custom formatting of tables and reports to be sent back. This gives the AMA significant flexibility to operate over challenged networks, both providing new degrees of freedom over existing configuration based data models used in synchronous networks and allowing for more concise formatting over constrained networks. This architecture makes very few assumptions on the nature of the network and allow for

continuous operation through periods of connectivity and lack of connectivity. The AMA deviates from synchronous management approaches because it never requires periods of bi-directional connectivity, and provides the manager flexibility to describe agent behavior that was unpredicted at the time of the data model creation.

To understand the unique motivations for the architecture, this section discusses motivating characteristics of challenged networks, current network management approaches, and how they might behave in a challenged environment.

3.1. Challenged Networks

A challenged network is one that "has serious trouble maintaining what an application would today expect of the end-to-end IP model" ([RFC7228]). This includes cases where there is never simultaneous end-to-end connectivity, when such connectivity is interrupted at planned or unplanned intervals, or when delays exceed those that could be accommodated by IP-based transport. Links in such networks are often unavailable due to attenuations, propagation delays, mobility, occultation, and other limitations imposed by energy and mass considerations.

Challenged networks exhibit the following properties that impact the way in which the function of network management is considered.

* No end-to-end path is guaranteed to exist at any given time between any two nodes.

* Round-trip communications between any two nodes within any given time window may be impossible.

* Latencies on the order of seconds, hours, or days must be tolerated.

* Links may be uni-directional.

* Bi-directional links may have asymmetric data rates.

One way in which constrained networks differ from challenged networks is the way in which the topology and, otherwise, roles and responsibilities of the network may evolve over time. From the time at which data is generated on a source node to the time at which the data is received at a destination node, the topology of the network may have changed. In certain circumstances, the physical node receiving messages for a given node identifier may also have changed.

When this topological change impacts the transport of messages, then transports must wait for the incremental connectivity necessary to advance messages along their expected route. Therefore, these networks cannot guarantee that there exist timely data exchange between managing and managed devices. For example, the Bundle Protocol transport protocol for use in DTNs implements this type of store-and-forward operation.

When topological change impacts the semantic roles and responsibilities of nodes in the network, then local configuration and autonomy at nodes must be present to determine time-variant changes. For example, the BPSec protocol does not encode security destinations and, instead, requires nodes in a network to identify as verifiers or acceptors when receiving secured messages.

When applied to network management, the semantic roles of Agent and Manager may also change with the changing topology of the network. Individual nodes must implement desirable behavior without reliance on a single oracle of configuration or other coordinating function such as an operator-in-the-loop. This implies that there MUST NOT be a defined relationship between a particular manager and agent in a network.

3.2. Current Approaches and Their Limitations

Network management solutions have been prevalent for many years in both local-area and wide-area networks. These range from the simplistic ability to configure settings of operational devices or report on state and operational conditions; to the more more complex modeling of an entire managed device setting, state, and behavior, pushing and receiving large sets of configuration data between the manager and the agent. Autonomy has more recently been applied to network management but is focused more on well resourced, unchallenged networks where devices self-configure, self-heal, and self-optimize with other nodes within their vicinity. This section describes some of the well known standardized protocols for network management as well as various proposed solutions and aims to differentiate their purpose with the needs of challenged network management solutions.

3.2.1. Simple Network Management Protocol (SNMP)

Historically, network management tools in unchallenged networks provide mechanisms for communicating locally-collected data from devices to operators and managing applications, typically using a "pull" mechanism where data must be explicitly requested by a Manager in order to be transmitted by an Agent. A legacy method for management in unchallenged networks today is the Simple Network

Management Protocol (SNMP) [RFC3416]. SNMP utilizes a request/ response model to set and retrieve data values such as host identifiers, link utilizations, error rates, and counters between application software on Agents and Managers. Data may be directly sampled or consolidated into representative statistics. Additionally, SNMP supports a model for asynchronous notification messages, called traps, based on predefined triggering events. Thus, Managers can query Agents for status information, send new configurations, and be informed when specific events have occurred. Traps and queryable data are defined in one or more Managed Information Bases (MIBs) which define the information for a particular data standard, protocol, device, or application.

While there is a large installation base for SNMP there are several aspects of the protocol that make in inappropriate for use in a challenged networking environment. SNMP relies on sessions with low round-trip latency to support its "pull" model. Complex management can be achieved but only through craftful orchestration using a series of real-time manager generated query and response logic not possible in challenged networks. The SNMP trap model provides some Agent-side processing, however because the processing has very low fidelity and traps are typically "fire and forget." Adaptive modifications to SNMP to support challenged networks and more complex application-level management, would alter the basic function of the protocol (data models, control flows, and syntax) so as to be functionally incompatible with existing SNMP installations. Therefore, this approach is not suitable for an asynchronous network management system.

3.2.2. YANG, NETCONF, and RESTCONF

Yet Another Next Generation (YANG) [RFC6020] is a data modeling language used to model configuration and state data of managed devices and applications. The YANG model defines a schema for organizing and accessing a device's configuration or operational information. Once a model is developed, it is loaded to both the client (manager) and server (agent) and serves as a contract between the two. A YANG model can be complex, describing many containers of managed elements, each with many configuration or operational state data nodes. It can further define lists of like elements. YANG allows for the definition of parameterized Remote Procedure Calls (RPCs) to be executed on managed nodes as well as the definition of asynchronous notifications within the model.

YANG by itself serves no purpose other than to organize data and describe the allowed configuration parameters on the managed device. The Network Configuration Protocol (NETCONF) [RFC6241] and the RESTCONF protocol [RFC8040] provide the mechanisms to install,

manipulate, and delete the configuration of network devices, using the YANG modules. NETCONF is a stateful, XML-based protocol that provides the RPC syntax to retrieve, edit, copy, or delete any data nodes or exposed functionality on the server. NETCONF connections are required to provide authentication, data integrity, confidentiality, and replay protection through secure transport protocols such as SSH or TLS. RESTCONF is a stateless RESTful protocol based on HTTP that uses JSON encoding to GET, POST, PUT, PATCH, or DELETE data nodes within the YANG modules similar to NETCONF. RESTCONF, while stateless, still requires secure transport such as TLS. Both NETCONF and RESTCONF place no specific functional requirements or constraints on the capabilities of the server, which makes it a very flexible tool for configuring a homogeneous network of devices, however they are limiting in challenged networks due to their requirements of underlying transport and dependence on the YANG data models.

NETCONF places specific constraints on any underlying transport protocol: a long-lived, reliable, low-latency sequenced data delivery session. No data is transferred without first establishing this bi-directional NETCONF session. RESTCONF relaxes this constraint however is limited to requesting or configuring individual data elements or entire containers within the YANG data model. It is therefore quite verbose and limited by the structure previously defined in the YANG module and any autonomous behavior depends on client slide orchestration similar to SNMP.

As previously noted, YANG allows for the definition of RPCs within the model and notification elements for asynchronous messaging. The RPCs provide both the definition of input and output parameters however are strictly allowed in NETCONF and RESTCONF to be sent as sequential procedures. Even if multiple procedures are sent, the server is required to execute them and reply in the order they were received. There is also no flexibility for the state-based execution of those procedures on the server. The RPCs are executed as soon as they are received, ultimately limiting the degrees of autonomy of the server. YANG notifications are quite promising for asynchronous network management, defined as both subscriptions to YANG notifications [RFC8639] and YANG PUSH notifications [RFC8641]. Notification containers must first be defined within the YANG module declaring the containers or data nodes of interest. The events can be filtered according to XPATH filtering defined in [RFC8639] Section 6, however generation of events are streamed and generally limited to the external changing state of a data node. YANG PUSH allows for both periodic and on-change event notification but supports no rules-based triggering. While the YANG data model offers many great features, the features today are simply limiting for the autonomous behavior required by challenged network management.

YANG is additionally limiting for challenged networks because of its non-hierarchal schema. While the YANG model flexibility is great for the management of nodes and applications of any type in an unchallenged network, it becomes a burden in challenged networks where concise encoding is necessary. All the data nodes within a YANG model are referenced by verbose string based path of the module, sub-module, container, and any data nodes such as lists, leaf-lists, or leafs. Recent efforts are underway which allow for CBOR encoding of YANG models [I-D.ietf-core-yang-cbor] and addressing of data nodes through integer value YANG Schema Item iDentifiers (SIDs) [I-D.ietf-core-sid], however these lack any formal hierarchal structure. All mapping of SIDs to YANG modules and data nodes is preformed manually which limits the portability of models and further increases the size of any encoding scheme.

3.2.3. Constrained RESTful Network Management

Due to the advent and ubiquity of the Internet of Things (IoT), the Constrained Application Protocol (CoAP) [RFC7252] has been recently developed for communicating with nodes and applications in constrained networks. CoAP is merely the messaging framework designed to limit message size and fragmentation, operating over IP networks. Because constrained networks could experience interruption similar to those in DTNs, the protocol provides for application layer store-and-forward as well as proxy delivery of messages, but is bound to UDP transport. An approach to network management has been authored that uses CoAP for transport and YANG as the data model, and is defined as CORECONF [I-D.ietf-core-comi]. This proposed protocol makes use of the YANG to CBOR encoding including the use of SIDs to limit message size, however is currently bound to UDP/IP transport of CoAP and further defines security requirements including DTLS or OSCORE. This explicit binding to transport and security protocols is limiting when applied to novel DTN approaches designed for challenged networks.

4. Services Provided by an AMA

This section identifies the services that an AMA would provide for management of challenged network resources. These services include configuration, reporting, parameterized control, and administration.

4.1. Configuration

Configuration services update Agent data associated with managed applications and protocols. Some configuration data might be defined in the context of an application or protocol, such that any network using that application or protocol would understand that data. Other configuration data may be defined tactically for use in a specific network deployment and not available to other networks even if they use the same applications or protocols.

New configurations received by an Agent must be validated to ensure that they do not conflict with other configurations or would otherwise prevent the Agent from effectively working with other Actors in its region. With no guarantee of round-trip data exchange, Agents cannot rely on remote Managers to correct erroneous or stale configurations from harming the flow of data through a challenged network.

Examples of configuration service behavior include the following.

* Creating a new datum as a function of other well-known data:

  C = A + B.

* Creating a new report as a unique, ordered collection of known data:

  RPT = {A, B, C}.

* Storing predefined, parameterized responses to potential future conditions:

  IF (X > 3) THEN RUN CMD(PARM).

4.2. Reporting

Reporting services populate report templates with values collected or computed by an Agent. The resultant reports are sent to one or more Managers by the Agent. The term "reporting" is used in place of the term "monitoring", as monitoring implies a timeliness and regularity that cannot be guaranteed by a challenged network. Reports sent by an Agent provide best-effort information to receiving Managers.

Since a Manager is not actively "monitoring" an Agent, the Agent must make its own determination on when to send what Reports based on its own local time and state information. Agents should produce Reports of varying fidelity and with varying frequency based on thresholds and other information set as part of configuration services.

Examples of reporting service behavior include the following.

* Generate Report R1 every hour (time-based production).

* Generate Report R2 when X > 3 (state-based production).

4.3. Autonomous Parameterized Procedure Calls

Similar to an RPC call, some mechanism MUST exist which allows a procedure to be run on an Agent in order to affect its behavior or otherwise change its internal state. Since there is no guarantee that a Manager will be in contact with an Agent at any given time, the decisions of whether and when a procedure should be run MUST be made locally and autonomously by the Agent. Two types of automation triggers are identified in the AMA: triggers based on the internal state of the Agent and triggers based on an Agent's notion of time. As such, the autonomous execution of procedures can be viewed as a stimulus-response system, where the stimulus is the positive evaluation of a state or time based predicate and the response is the function to be executed.

The autonomous nature of procedure execution by an Agent implies that the full suite of information necessary to run a procedure may not be known by a Manager in advance. To address this situation, a parameterization mechanism MUST be available so that required data can be provided at the time of execution on the Agent rather than at the time of definition/configuration by the Manager.

Autonomous, parameterized procedure calls provide a powerful mechanism for Managers to "manage" an Agent asynchronously during periods of no communication by pre-configuring responses to events that may be encountered by the Agent at a future time.

Examples of potential behavior include the following.

* Updating local routing information based on instantaneous link analysis.

* Managing storage on the device to enforce quotas.

* Applying or modifying local security policy.

4.4. Administration

Administration services enforce the potentially complex mapping of configuration, reporting, and control services amongst Agents and Managers in the network. Fine-grained access controls that specify which Managers may apply which services to which Agents may be necessary in networks that either deal with multiple administrative entities or overlay networks that cross administrative boundaries. Whitelists, blacklists, key-based infrastructures, or other schemes may be used for this purpose.

Examples of administration service behavior include the following.

* Agent A1 only Sends reports for Protocol P1 to Manager M1.

* Agent A2 only accepts a configurations for Application Y from Managers M2 and M3.

* Agent A3 accepts services from any Manager providing the proper authentication token.

Note that the administrative enforcement of access control is different from security services provided by the networking stack carrying such messages.

5. Desirable Properties of an AMA

This section describes those design properties that are desirable when defining an architecture that must operate across challenged links in a network. These properties ensure that network management capabilities are retained even as delays and disruptions in the network scale. Ultimately, these properties are the driving design principles for the AMA.

5.1. Intelligent Push of Information

Pull management mechanisms require that a Manager send a query to an Agent and then wait for the response to that query. This practice implies a control-session between entities and increases the overall message traffic in the network. Challenged networks cannot guarantee that the round-trip data-exchange will occur in a timely fashion. In extreme cases, networks may be comprised of solely uni-directional links which drastically increases the amount of time needed for a round-trip data exchange. Therefore, pull mechanisms must be avoided in favor of push mechanisms.

Push mechanisms, in this context, refer to the ability of Agents to leverage rule-based criteria to determine when and what information should be sent to managers. This could be based solely off logic applied to existing VARs or EDDs, or based off operations applied to data elements. Such mechanisms do not require round-trip communications as Managers do not request each reporting instance; Managers need only request once, in advance, that information be produced in accordance with a predetermined schedule or in response to a predefined state on the Agent. In this way information is "pushed" from Agents to Managers and the push is "intelligent" because it is based on some internal evaluation performed by the Agent.

5.2. Minimize Message Size Not Node Processing

Protocol designers must balance message size versus message processing time at sending and receiving nodes. Verbose representations of data simplify node processing whereas compact representations require additional activities to generate/parse the compacted message. There is no asynchronous management advantage to minimizing node processing time in a challenged network. However, there is a significant advantage to smaller message sizes in such networks. Compact messages require smaller periods of viable transmission for communication, incur less re-transmission cost, and consume less resources when persistently stored en-route in the network. An Asynchronous Management Protocol (AMP) should minimize PDUs whenever practical, to include packing and unpacking binary data, variable-length fields, and pre-configured data definitions.

5.3. Absolute Data Identification

Elements within the management system must be uniquely identifiable so that they can be individually manipulated. Identification schemes that are relative to system configuration make data exchange between Agents and Managers difficult as system configurations may change faster than nodes can communicate.

Consider the following common technique for approximating an associative array lookup. A manager wishing to do an associative lookup for some key K1 will (1) query a list of array keys from the agent, (2) find the key that matches K1 and infer the index of K1 from the returned key list, and (3) query the discovered index on the agent to retrieve the desired data.

Ignoring the inefficiency of two pull requests, this mechanism fails when the Agent changes its key-index mapping between the first and second query. Rather than constructing an artificial mapping from K1 to an index, an AMP must provide an absolute mechanism to lookup the value K1 without an abstraction between the Agent and Manager.

5.4. Custom Data Definition

Custom definition of new data from existing data (such as through data fusion, averaging, sampling, or other mechanisms) provides the ability to communicate desired information in as compact a form as possible. Specifically, an Agent should not be required to transmit a large data set for a Manager that only wishes to calculate a smaller, inferred data set. These new defined data elements could be calculated and used both as parameters for local stimulus-response rules-based criteria or simply serve to populate custom reports and tables. Since the identification of custom data sets is likely to occur in the context of a specific network deployment, AMPs must provide a mechanism for their definition.

Aggregation of controls and custom formatting of reports and tables are is equally important. Custom reporting provides the flexibility allowing the manager to define the desired format of all information to be sent over the challenged network from the agents, serving to both save link capacity and increase the value of returned information. Aggregation of controls allows a manager to specify a set of controls to execute, specifying both the order and criteria of execution. This aggregate set of controls can be sent as a single command rather than a series of sequential operands. In this case it is additionally possible to use outputs of one command to serve as an input to the next at the agent.

5.5. Autonomous Operation

AMA network functions must be achievable using only knowledge local to the Agent. Rather than directly controlling an Agent, a Manager configures an engine of the Agent to take its own action under the appropriate conditions in accordance with the Agent's notion of local state and time.

Such an engine may be used for simple automation of predefined tasks or to support semi-autonomous behavior in determining when to run tasks and how to configure or parameterize tasks when they are run. Wholly autonomous operations MAY be supported where required. Generally, autonomous operations should provide the following benefits.

* Distributed Operation - The concept of pre-configuration allows the Agent to operate without regular contact with Managers in the system. The initial configuration (and periodic update) of the system remains difficult in a challenged network, but an initial synchronization on stimuli and responses drastically reduces needs for centralized operations.

* Deterministic Behavior - Such behavior is necessary in critical operational systems where the actions of a platform must be well understood even in the absence of an operator in the loop. Depending on the types of stimuli and responses, these systems may be considered to be maintaining simple automation or semi-autonomous behavior. In either case, this preserves the ability of a frequently-out-of-contact Manager to predict the state of an Agent with more reliability than cases where Agents implement independent and fully autonomous systems.

* Engine-Based Behavior - Several operational systems are unable to deploy "mobile code" based solutions due to network bandwidth, memory or processor loading, or security concerns. Engine-based approaches provide configurable behavior without incurring these types of concerns associated with mobile code.

6. AMA Roles and Responsibilities

By definition, Agents reside on managed devices and Managers reside on managing devices. There is however no pre-supposed architecture that connects managers and agents and therefore a single device could assume both roles. This section describes the responsibilities associated with each role and how these roles participate in network management.

6.1. Agent Responsibilities

Application Support
: Agents MUST collect all data, execute all procedures, populate all reports and run operations required by each application which the Agent manages. Agents MUST report supported applications so that Managers in a network understands what information is understood by what Agent.

Local Data Collection
: Agents MUST collect from local firmware (or other on-board mechanisms) and report all data defined for the management of applications for which they have been configured.

Autonomous Control
Agents MUST determine, as previously prescribed by a manager, whether a procedure should be invoked.

User Data Definition
Agents MUST provide mechanisms for operators in the network to use configuration services to create customized data definitions in the context of a specific network or network use-case. Agents MUST allow for the creation, listing, and removal of such definitions in accordance with whatever security models are deployed within the particular network.

Where applicable, Agents MUST verify the validity of these definitions when they are configured and respond in a way consistent with the logging/error-handling policies of the Agent and the network.

Autonomous Reporting
Agents MUST determine, without real-time Manager intervention, whether and when to populate and transmit a given report targeted to one or more Managers in the network.

Consolidate Messages
Agents SHOULD produce as few messages as possible when sending information. For example, rather than sending multiple messages, each with one report to a Manager, an Agent SHOULD prefer to send a single message containing multiple reports.

6.2. Manager Responsibilities

Agent Capabilities Mapping
Managers MUST understand what applications are managed by the various Agents with which they communicate. Managers should not attempt to request, invoke, or refer to application information for applications not managed by an Agent.

Data Collection
Managers MUST receive information from Agents by asynchronously configuring the production of reports and then waiting for, and collecting, responses from Agents over time. Managers MAY try to detect conditions where Agent information has not been received within operationally relevant time spans and react in accordance with network policy.

Custom Definitions
Managers should provide the ability to define custom data definitions. Any custom definitions MUST be transmitted to

appropriate Agents and these definitions MUST be remembered to interpret the reporting of these custom values from Agents in the future.

Data Translation
Managers should provide some interface to other network management protocols. Managers MAY accomplish this by accumulating a repository of push-data from high-latency parts of the network from which data may be pulled by low-latency parts of the network.

Data Fusion
Managers MAY support the fusion of data from multiple Agents with the purpose of transmitting fused data results to other Managers within the network. Managers MAY receive fused reports from other Managers pursuant to appropriate security and administrative configurations.

7. Logical Data Model

The AMA logical data model captures the types of information that should be collected and exchanged to implement necessary roles and responsibilities. The data model presented in this section does not presuppose a specific mapping to a physical data model or encoding technique; it is included to provide a way to logically reason about the types of data that should be exchanged in an asynchronously managed network.

The elements of the AMA logical data model are described as follows.

7.1. Data Representations: Constants, Externally Defined Data, and Variables

There are three fundamental representations of data in the AMA: (1) data whose values do not change as a function of time or state, (2) data whose values change as determined by sampling/calculation external to the network management system, and (3) data whose values are calculated internal to the network management system.

Data whose values do not change as a function of time or state are defined as Constants (CONST). CONST values are strongly typed, named values that cannot be modified once they have been defined.

Data sampled/calculated external to the network management system are defined as Externally Defined Data" (EDD). EDD values represent the most useful information in the management system as they are provided by the applications or protocols being managed on the Agent. It is RECOMMENDED that EDD values be strongly typed to avoid issues with

interpreting the data value. It is also RECOMMENDED that the timeliness/staleness of the data value be considered when using the data in the context of autonomous action on the Agent.

Data that is calculated internal to the network management system is defined as a Variable (VAR). VARs allow the creation of new data values for use in the network management system. New value definitions are useful for storing user-defined information, storing the results of complex calculations for easier re-use, and providing a mechanism for combining information from multiple external sources. It is RECOMMENDED that VARs be strongly typed to avoid issues with interpreting the data value. In cases where a VAR definition relies on other VAR definitions, mechanisms to prevent circular references MUST be included in any actual data model or implementation.

7.2. Data Collections: Reports and Tables

Individual data values may be exchanged amongst Agents and Managers in the AMA. However, data are typically most useful to a Manager when received as part of a set of information. Ordered collections of data values can be produced by Agents and sent to Managers as a way of efficiently communicating Agent status. Within the AMA, the structure of the ordered collection is treated separately from the values that populate such a structure.

The AMA provides two ways of defining collections of data: reports and tables. Reports are ordered sets of data values, whereas Tables are special types of reports whose entries have a regular, tabular structure.

7.2.1. Report Templates and Reports

The typed, ordered structure of a data collection is defined as a Report Template (RPTT). A particular set of data values provided in compliance with such a template is called a Report (RPT).

Separating the structure and content of a report reduces the overall size of RPTs in cases where reporting structures are well known and unchanging. RPTTs can be synchronized between an Agent and a Manager so that RPTs themselves do not incur the overhead of carrying self-describing data. RPTTs may include EDD values, VARs, and also other RPTTs. In cases where a RPTT includes another RPTTs, mechanisms to prevent circular references MUST be included in any actual data model or implementation.

Protocols and applications managed in the AMA may define common RPTTs. Additionally, users within a network may define their own RPTTs that are useful in the context of a particular deployment.

Unlike tables, reports do not exploit assumptions on the underlying structure of their data. Therefore, unlike tables, operators can define new reports at any time as part of the runtime configuration of the network.

7.2.2. Table Templates and Tables

Tables optimize the communication of multiple sets of data in situations where each data set has the same syntactic structure and with the same semantic meaning. Unlike reports, the regularity of tabular data representations allow for the addition of new rows without changing the structure of the table. Attempting to add a new data set at the end of a report would require alterations to the report template.

The typed, ordered structure of a table is defined as a Table Template (TBLT). A particular instance of values populating the table template is called a Table (TBL).

TBLTs describes the "columns" that define the table schema. A TBL represents the instance of a specific TBLT that holds actual data values. These data values represent the "rows" of the table.

The prescriptive nature of the TBLT allows for the possibility of advanced filtering which may reduce traffic between Agents and Managers. However, the unique structure of each TBLT may make them difficult or impossible to change dynamically in a network.

7.3. Command Execution: Controls and Macros

Low-latency, high-availability approaches to network management use mechanisms such as (or similar to) RPCs to cause some action to be performed on an Agent. The AMA enables similar capabilities without requiring that the Manager be in the processing loop of the Agent. Command execution in the AMA happens through the use of controls and macros.

A Control (CTRL) represents a parameterized, predefined procedure that can be run on an Agent. While conceptually similar to a "remote procedure call", CTRLs differ in that they do not provide numeric return codes. The concept of a return code when running a procedure implies a synchronous relationship between the caller of the procedure and the procedure being called, which is disallowed in an asynchronous management system. Instead, CTRLs may create reports which describe the status and other summarizations of their operation, and these reports may be sent to the Manager(s) calling the CTRL.

Parameters can be provided when running a command from a Manager, pre-configured as part of a response to a time-based or state-based rule on the Agent, or auto-generated as needed on the Agent. The success or failure of a control MAY be inferred by reports generated for that purpose.

NOTE: The AMA term control is derived in part from the concept of Command and Control (C2) where control implies the operational instructions that must be undertaken to implement (or maintain) a commanded objective. An asynchronous management function controls an Agent to allow it to fulfill its commanded purpose in a variety of operational scenarios. For example, attempting to maintain a safe internal thermal environment for a spacecraft is considered "thermal control" (not "thermal commanding") even though thermal control involves "commanding" heaters, louvers, radiators, and other temperature-affecting components.

Often, a series of controls must be executed in sequence to achieve a particular outcome. A Macro (MACRO) represents an ordered collection of controls (or other macros). In cases where a MACRO includes another MACRO, mechanisms to prevent circular references and maximum nesting levels MUST be included in any actual data model or implementation.

7.4. Autonomy: Time and State-Based Rules

The AMA data model contains EDDs and VARs that capture the state of applications on an Agent. The model also contains controls and macros to perform actions on an Agent. A mechanism is needed to relate these two capabilities: to perform an action on the Agent in response to the state of the Agent. This mechanism in the AMA is the "rule" and can be activated based on Agent internal state (state-based rule) or based on the Agent's notion of relative time (time-based rule).

7.4.1. State-Based Rule (SBR)

State-Based Rules (SBRs) perform actions based on the Agent's internal state, as identified by EDD and VAR values. An SBR represents a stimulus-response pairing in the following form: IF predicate THEN response The predicate is a logical expression that evaluates to true if the rule stimulus is present and evaluates to false otherwise. The response may be any control or macro known to the Agent.

An example of an SBR could be to turn off a heater if some internal temperature is greater than a threshold: IF (current_temp > maximum_temp) THEN turn_heater_off

Rules may construct their stimuli from the full set of values known to the network management system. Similarly, responses may be constructed from the full set of controls and macros that can be run on the Agent. By allowing rules to evaluate the variety of all known data and run the variety of all known controls, multiple applications can be monitored and managed by one (or few) Agent instances.

7.4.2. Time-Based Rule (TBR)

Time-Based Rules (TBR) perform actions based on the Agent's notion of the passage of time. A possible TBR construct would be to perform some action at 1Hz on the Agent.

A TBR is a specialization of an SBR as the Agent's notion of time is a type of Agent state. For example, a TBR to perform an action every 24 hours could be expressed using some type of predicate of the form: IF (((current_time - base_time) % 24_hours) == 0) THEN ... However, time-based events are popular enough that special semantics for expressing them would likely significantly reduce the computations necessary to represent time functions in a SBR.

7.5. Calculations: Expressions, Literals, and Operators

Actions such as computing a VAR value or describing a rule predicate require some mechanism for calculating the value of mathematical expressions. In addition to the aforementioned AMA logical data objects, Literals, Operators, and Expressions are used to perform these calculations.

A Literal (LIT) represents a strongly typed datum whose identity is equivalent to its value. An example of a LIT value is "4" - its identifier (4) is the same as its value (4). Literals differ from constants in that constants have an identifier separate from their value. For example, the constant PI may refer to a value of 3.14. However, the literal 3.14159 always refers to the value 3.14159.

An Operator (OP) represents a mathematical operation in an expression. OPs should support multiple operands based on the operation supported. A common set of OPs SHOULD be defined for any Agent and systems MAY choose to allow individual applications to define new OPs to assist in the generation of new VAR values and predicates for managing that application. OPs may be simple binary operations such as "A + B" or more complex functions such as sin(A) or avg(A,B,C,D). Additionally, OPs may be typed. For example, addition of integers may be defined separately from addition of real numbers.

An Expression (EXPR) is a combination of operators and operands used to construct a numerical value from a series of other elements of the AMA logical model. Operands include any AMA logical data model object that can be interpreted as a value, such as EDD, VAR, CONST, and LIT values. Operators perform some function on operands to generate new values.

8. System Model

This section describes the notional data flows and control flows that illustrate how Managers and Agents within an AMA cooperate to perform network management services.

8.1. Control and Data Flows

The AMA identifies three significant data flows: control flows from Managers to Agents, reports flows from Agents to Managers, and fusion reports from Managers to other Managers. These data flows are illustrated in Figure 1.

AMA Control and Data Flows

```
       +---------+       +------------------------+      +---------+
       | Node A  |       |         Node B         |      |  Node C |
       |         |       |                        |      |         |
       |+-------+|       |+-------+      +-------+|      |+-------+|
       ||       ||=====>>||Manager|====>>|       ||====>>||       ||
       ||       ||<<=====||   B   |<<====|Agent B||<<====||       ||
       ||       ||       |+--++---+      +-------+|      ||Manager||
       || Agent ||       +---||-------------------+      ||   C   ||
       ||   A   ||           ||                          ||       ||
       ||       ||<<=========||==========================||       ||
       ||       ||===========++========================>>||       ||
       |+-------+|                                       |+-------+|
       +---------+                                       +---------+
```

Figure 1

In this data flow, the Agent on node A receives Controls from Managers on nodes B and C, and replies with Report Entries back to these Managers. Similarly, the Agent on node B interacts with the local Manager on node B and the remote Manager on node C. Finally, the Manager on node B may fuse Report Entries received from Agents at nodes A and B and send these fused Report Entries back to the Manager on node C. From this figure it is clear that there exist many-to-many relationships amongst Managers, amongst Agents, and between Agents and Managers. Note that Agents and Managers are roles, not

necessarily different software applications. Node A may represent a single software application fulfilling only the Agent role, whereas node B may have a single software application fulfilling both the Agent and Manager roles. The specifics of how these roles are realized is an implementation matter.

8.2. Control Flow by Role

This section describes three common configurations of Agents and Managers and the flow of messages between them. These configurations involve local and remote management and data fusion.

8.2.1. Notation

The notation outlined in Table 1 describes the types of control messages exchanged between Agents and Managers.

Term	Definition	Example
EDD#	EDD definition.	EDD1
V#	Variable definition.	V1 = EDD1 + V0.
DEF([ACL], ID,EXPR)	Define ID from expression. Allow managers in access control list (ACL) to request this ID.	DEF([*], V1, EDD1 + EDD2)
PROD(P,ID)	Produce ID according to predicate P. P may be a time period (1s) or an expression (EDD1 > 10).	PROD(1s, EDD1)
RPT(ID)	A report identified by ID.	RPT(EDD1)

Table 1: Terminology

8.2.2. Serialized Management

This is a nominal configuration of network management where a Manager interacts with a set of Agents. The control flows for this are outlined in Figure 2.

Serialized Management Control Flow

```
         +----------+            +---------+           +---------+
         |  Manager |            | Agent A |           | Agent B |
         +----+-----+            +----+----+           +----+----+
              |                       |                     |
              |-----PROD(1s, EDD1)--->|                     | (1)
              |----------------------------PROD(1s, EDD1)-->|
              |                       |                     |
              |                       |                     |
              |<-------RPT(EDD1)------|                     | (2)
              |<----------------------------RPT(EDD1)-------|
              |                       |                     |
              |                       |                     |
              |<-------RPT(EDD1)------|                     |
              |<----------------------------RPT(EDD1)-------|
              |                       |                     |
              |                       |                     |
              |<-------RPT(EDD1)------|                     |
              |<----------------------------RPT(EDD1)-------|
              |                       |                     |
```

Figure 2

In a simple network, a Manager interacts with multiple Agents.

In this figure, the Manager configures Agents A and B to produce EDD1 every second in (1). Upon receiving and configuring this message, Agents A and B then build a Report Entry containing EDD1 and send those reports back to the Manager in (2). This behavior then repeats this action every 1s without requiring other inputs from the Manager.

8.2.3. Multiplexed Management

Networks spanning multiple administrative domains may require multiple Managers (for example, one per domain). When a Manager defines custom Reports/Variables to an Agent, that definition may be tagged with an Access Control List (ACL) to limit what other Managers will be privy to this information. Managers in such networks should synchronize with those other Managers granted access to their custom data definitions. When Agents generate messages, they MUST only send messages to Managers according to these ACLs, if present. The control flows in this scenario are outlined in Figure 3.

Multiplexed Management Control Flow

```
        +-----------+            +-------+            +-----------+
        | Manager A |            | Agent |            | Manager B |
        +-----+-----+            +---+---+            +-----+-----+
              |                      |                      |
              |---DEF(A,V1,EDD1*2)-->|<-DEF(B, V2, EDD2*2)--| (1)
              |                      |                      |
              |---PROD(1s, V1)------>|<---PROD(1s, V2)------| (2)
              |                      |                      |
              |<--------RPT(V1)------|                      | (3)
              |                      |--------RPT(V2)------>|
              |<--------RPT(V1)------|                      |
              |                      |--------RPT(V2)------>|
              |                      |                      |
              |                      |<---PROD(1s, V1)------| (4)
              |                      |                      |
              |                      |---ERR(V1 no perm.)-->|
              |                      |                      |
              |--DEF(*,V3,EDD3*3)--->|                      | (5)
              |                      |                      |
              |---PROD(1s, V3)------>|                      | (6)
              |                      |                      |
              |                      |<----PROD(1s, V3)-----|
              |                      |                      |
              |<--------RPT(V3)------|--------RPT(V3)------>| (7)
              |<--------RPT(V1)------|                      |
              |                      |--------RPT(V2)------>|
              |<-------RPT(V3)-------|--------RPT(V3)------>|
              |<-------RPT(V1)-------|                      |
              |                      |--------RPT(V2)------>|
```

Figure 3

Complex networks require multiple Managers interfacing with Agents.

In more complex networks, any Manager may choose to define custom Reports and Variables, and Agents may need to accept such definitions from multiple Managers. Variable definitions may include an ACL that describes who may query and otherwise understand these definitions. In (1), Manager A defines V1 only for A while Manager B defines V2 only for B. Managers may, then, request the production of Report Entries containing these definitions, as shown in (2). Agents produce different data for different Managers in accordance with configured production rules, as shown in (3). If a Manager requests the production of a custom definition for which the Manager has no permissions, a response consistent with the configured logging policy on the Agent should be implemented, as shown in (4). Alternatively, as shown in (5), a Manager may define custom data with no access restrictions, allowing all other Managers to request and use this

definition. This allows all Managers to request the production of Report Entries containing this definition, shown in (6) and have all Managers receive this and other data going forward, as shown in (7).

8.2.4. Data Fusion

Data fusion reduces the number and size of messages in the network which can lead to more efficient utilization of networking resources. The AMA supports fusion of NM reports by co-locating Agents and Managers on nodes and offloading fusion activities to the Manager. This process is illustrated in Figure 4.

Data Fusion Control Flow

```
+-----------+        +-----------+      +---------+      +---------+
| Manager A |        | Manager B |      | Agent B |      | Agent C |
+---+-------+        +-----+-----+      +----+----+      +----+----+
    |                      |                 |                |
    |-DEF(A,V0,EDD1+EDD2)->|                 |                | (1)
    |-PROD(EDD1&EDD2,V0)-->|                 |                |
    |                      |                 |                |
    |                      |--PROD(1s,EDD1)->|                | (2)
    |                      |------------------PROD(1s, EDD2)->|
    |                      |                 |                |
    |                      |<---RPT(EDD1)----|                | (3)
    |                      |<------------------RPT(EDD2)------|
    |                      |                 |                |
    |<-----RPT(A,V0)-------|                 |                | (4)
    |                      |                 |                |
```

Figure 4

Data fusion occurs amongst Managers in the network.

In this example, Manager A requires the production of a Variable V0, from node B, as shown in (1). The Manager role understands what data is available from what agents in the subnetwork local to B, understanding that EDD1 is available locally and EDD2 is available remotely. Production messages are produced in (2) and data collected in (3). This allows the Manager at node B to fuse the collected Report Entries into V0 and return it in (4). While a trivial example, the mechanism of associating fusion with the Manager function rather than the Agent function scales with fusion complexity, though it is important to reiterate that Agent and Manager designations are roles, not individual software components. There may be a single software application running on node B implementing both Manager B and Agent B roles.

9. IANA Considerations

This protocol has no fields registered by IANA.

10. Security Considerations

Security within an AMA MUST exist in two layers: transport layer security and access control.

Transport-layer security addresses the questions of authentication, integrity, and confidentiality associated with the transport of messages between and amongst Managers and Agents in the AMA. This security is applied before any particular Actor in the system receives data and, therefore, is outside of the scope of this document.

Finer grain application security is done via ACLs which are defined via configuration messages and implementation specific.

11. Informative References

[BIRRANE1] Birrane, E.B. and R.C. Cole, "Management of Disruption-Tolerant Networks: A Systems Engineering Approach", 2010.

[BIRRANE2] Birrane, E.B., Burleigh, S.B., and V.C. Cerf, "Defining Tolerance: Impacts of Delay and Disruption when Managing Challenged Networks", 2011.

[BIRRANE3] Birrane, E.B. and H.K. Kruse, "Delay-Tolerant Network Management: The Definition and Exchange of Infrastructure Information in High Delay Environments", 2011.

[I-D.ietf-core-comi] Veillette, M., Stok, P. V. D., Pelov, A., Bierman, A., and I. Petrov, "CoAP Management Interface (CORECONF)", Work in Progress, Internet-Draft, draft-ietf-core-comi-11, 17 January 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-core-comi-11>.

[I-D.ietf-core-sid] Veillette, M., Pelov, A., Petrov, I., and C. Bormann, "YANG Schema Item iDentifier (YANG SID)", Work in Progress, Internet-Draft, draft-ietf-core-sid-16, 24 June 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-core-sid-16>.

[I-D.ietf-core-yang-cbor]
Veillette, M., Petrov, I., Pelov, A., and C. Bormann, "CBOR Encoding of Data Modeled with YANG", Work in Progress, Internet-Draft, draft-ietf-core-yang-cbor-16, 24 June 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-core-yang-cbor-16>.

[I-D.ietf-dtn-bpbis]
Burleigh, S., Fall, K., and E. J. Birrane, "Bundle Protocol Version 7", Work in Progress, Internet-Draft, draft-ietf-dtn-bpbis-31, 25 January 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-dtn-bpbis-31>.

[I-D.irtf-dtnrg-dtnmp]
Birrane, E. J. and V. Ramachandran, "Delay Tolerant Network Management Protocol", Work in Progress, Internet-Draft, draft-irtf-dtnrg-dtnmp-01, 31 December 2014, <https://datatracker.ietf.org/doc/html/draft-irtf-dtnrg-dtnmp-01>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3416] Presuhn, R., Ed., "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3416, DOI 10.17487/RFC3416, December 2002, <https://www.rfc-editor.org/info/rfc3416>.

[RFC4838] Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, R., Scott, K., Fall, K., and H. Weiss, "Delay-Tolerant Networking Architecture", RFC 4838, DOI 10.17487/RFC4838, April 2007, <https://www.rfc-editor.org/info/rfc4838>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <https://www.rfc-editor.org/info/rfc7228>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Notifications", RFC 8639, DOI 10.17487/RFC8639, September 2019, <https://www.rfc-editor.org/info/rfc8639>.

[RFC8641] Clemm, A. and E. Voit, "Subscription to YANG Notifications for Datastore Updates", RFC 8641, DOI 10.17487/RFC8641, September 2019, <https://www.rfc-editor.org/info/rfc8641>.

Authors' Addresses

Edward J. Birrane
Johns Hopkins Applied Physics Laboratory

Email: Edward.Birrane@jhuapl.edu

Emery Annis
Johns Hopkins Applied Physics Laboratory

Email: Emery.Annis@jhuapl.edu

Sarah E. Heiner
Johns Hopkins Applied Physics Laboratory

Email: Sarah.Heiner@jhuapl.edu