OpenC2 Language Specification

Working Draft 0.0.2

09 Oct 2017

Technical Committee:
OASIS OpenC2 Technical Committee

Chair:
Joe Brule (jmbrule@nsa.gov), National Security Agency
Sounil Yu (sounil.yu@bankofamerica.com), Bank of America

Editors:

Related work:
This specification is related to:

Abstract:
Cyberattacks are increasingly sophisticated, less expensive to execute, dynamic and automated. The provision of cyberdefense via statically configured products operating in isolation is no longer tenable. Standardized interfaces, protocols and data models will facilitate the integration of the functional blocks within a system or enterprise. Open Command and Control (OpenC2) is a concise and extensible language to enable the command and control of cyber defense components, subsystems and/or systems in a manner that is agnostic of the underlying products, technologies, transport mechanisms or other aspects of the implementation. It should be understood that a language such as OpenC2 is necessary but insufficient to enable coordinated cyber response. Other aspects of coordinated cyber response such as sensing, analytics, and selecting appropriate courses of action are beyond the scope of OpenC2.

Status:
This Working Draft (WD) has been produced by one or more TC Members; it has not yet been voted on by the TC or approved as a Committee Draft (Committee Specification Draft or a Committee Note Draft). The OASIS document Approval Process begins officially with a TC vote to approve a WD as a Committee Draft. A TC may approve a Working Draft, revise it, and re-approve it any number of times as a Committee Draft.

URI patterns:
Initial publication URI:

Permanent “Latest version” URI:

(Managed by OASIS TC Administration; please don't modify.)
# TABLE OF CONTENTS

## Introduction
- Goal 6
- Purpose and Scope 6
- Terminology 6
- Document Conventions 7
  - 1.4.1 Naming Conventions 8
- Normative References 8
- Non normative References 8
- Conformance 8
- Acknowledgements 8

## OPENC2 LANGUAGE
- Overview 9
- OpenC2 Command 9
  - Command Structure 9
  - Action Vocabulary 10
  - Target Vocabulary 12
  - Actuator 12
  - Command Option Vocabulary 13
- OpenC2 Response 13
  - Response Structure 13

## OpenC2 Property Tables 6
LIST OF TABLES

Table 2-1. OpenC2 Command Field Descriptions 5
Table 2-2. Summary of Action Definitions 6
1. **INTRODUCTION**

The OpenC2 Specification defines a language used to compose messages that coordinate and execute the command and control of cyber defenses between and within networks and systems.

OpenC2 commands are composed of an action (what is to be done), a target (what is being acted upon), an optional actuator (what is executing the command), and optional command options, which influence how the command is to be performed.

A command that consists of an action coupled with a target is sufficient for a high level effects based command (e.g., mitigate evildomain.com). The extension of the command with an actuator and modifiers provides additional precision and specificity (e.g., deny ip=1.2.3.4 by actuator=firewall3 command-id=1eab14...). Additional detail about aspects of a command may be included to increase the precision of the command. For example, which target (i.e., target specifier), additional information about what is to be performed on a specific target type (i.e., target option), which actuator(s) (i.e., actuator specifier) and/or additional information regarding how a specific actuator executes the action (i.e., actuator option).

OpenC2 responses are issued as a result of a command. OpenC2 responses are used to provide acknowledgement, status, results of a query or other information in conjunction with a particular command.

OpenC2 alerts are similar to responses in that they provide state, status, or other information but are distinct from responses in that they are sent asynchronously (i.e., unsolicited and not necessarily associated with a particular command)

1.1 **Goal**

The goal of OpenC2 is to define a lexicon (language and semantics) at a level of abstraction that will enable the coordination and execution of command and control of cyber defenses between and within networks. The goal of this specification is to define the core components of a command such as the actions, a default namespace for the targets, a syntax, a lexicon and other aspects of a language that when coupled, create a complete command.

A command may be extended to include an actuator as defined in other OpenC2 specifications (i.e., actuator profiles) for additional precision. This accommodates the inclusion of new actuators (through other specifications) while providing a core specification that documents the syntax, actions, and targets.

1.2 **Purpose and Scope**

The OpenC2 Language Specification defines a baseline set of components to assemble a complete command and control message set and provide a framework so that the language can be extended to accommodate new technologies. To achieve this purpose, the scope of this specification includes:

1. the exclusive set of actions and options that may be used in OpenC2 commands,
2. the set of targets, target specifiers, and target options,
3. an organizational scheme that describes a foundational actuator profile.
4. a syntax to express commands and responses.
5. the default serialization of OpenC2 commands, and responses.
6. the procedures for extending the language to accommodate new technologies in a manner that is consistent with the OpenC2 Language Specification.

The OpenC2 language is necessary but insufficient for the realization of coordinated cyber response. Though necessary for cyber-response implementations, the following items are beyond the scope of this specification:

1. Language definitions for a particular actuator to extend the OpenC2 language. Extensions to the language will be captured in other specifications.
2. Specifying alternate serializations of OpenC2 commands. However, optional serializations may be documented in other specifications.
3. The enumeration of the protocols required for transport, information assurance, sensing, analytics and other external dependencies. The OpenC2 language assumes that the event has been detected, a decision to act has been made, the act is warranted, and the initiator and recipient of the commands are authenticated and authorized. The OpenC2 language was designed to be agnostic of the other aspects of cyber defense implementations that realize these assumptions.

1.3 Terminology
The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this specification are to be interpreted as described in IETF RFC 2119 [RFC2119].

1.4 Document Conventions
The following typographical conventions are used in this document.

*italics*
Indicates new terms, URLs, email addresses, filenames, and file extensions.

**ALL CAPS**
Used for components of the abstract syntax: ACTION, TARGET, ACTUATOR, MODIFIERS.

‘single quotes’
Used for action names.

<fields within single brackets>
Are replaced with the appropriate details.
1.4.1 Naming Conventions

All type names, property names and literals are in lowercase, except when referencing canonical names defined in another standard (e.g. literal values from an IANA registry). Words in property names are separated with an underscore (_), while words in type names and string enumerations are separated with a hyphen (-). All type names, property names, object names, and vocabulary terms are between three and 250 characters long.

```json
{ "action": "contain",
  "target": {
    "user_account": {
      "user_id": "fjbloggs",
      "account_type": "windows-local"}}}
```

1.5 Normative References

LIST OF NORMATIVE REFERENCES TBS

1.6 Non normative References

LIST OF NONNORMATIVE REFERENCES TBS

1.7 Conformance

OpenC2 is a command and control language that converges (i.e. common ‘point of understanding’) on a common syntax, and lexicon. OpenC2 does not have a dependency on a particular programming language, computing platform, transport protocol etc. Conformant implementations of OpenC2:

- MUST support OpenC2 commands, responses and alerts as defined in this document.
- MUST implement the actions designated as mandatory in this document.
- MUST implement the targets designated as mandatory in this document.
- MAY implement optional targets defined in this document.
- MAY implement actuator specifiers, actuator options, target specifiers and/or target options as specified in one or more actuator profiles.
- MUST implement JSON serialization of the commands, responses and alerts that are consistent with the syntax defined in this document.
- MAY implement other serializations of the commands, responses and alerts that are consistent with the syntax defined in this document.

1.8 Acknowledgements

TBS
2. **OPENC2 LANGUAGE**

2.1 **Overview**

The OpenC2 language has two distinct types of messages: *Command* and *Response*. At the most basic level, the OpenC2 Command describes an *action* performed on a *target*. The OpenC2 Response is used to provide execution status and optional data requested as a result of a command. OpenC2 Response messages may refer to the command that initiated the response.

2.2 **OpenC2 Command**

The OpenC2 Command communicates an action to be performed on a target and may include the entity that is to execute the command.

2.2.1 **Command Structure**

An OpenC2 command has four independently defined fields; ACTION, TARGET, ACTUATOR and COMMAND-OPTIONS:

The ACTION and TARGET fields are required and are populated by one of 31 ‘action-types’ and 23 ‘target-types’ which are defined in this specification. A particular target-type may be further refined by one or more ‘target-specifiers’ and/or ‘target-options’ which are also defined in this specification.

The ACTUATOR field (optional) identifies the entity(ies) that are tasked to execute the command. The ‘actuator-type’ is a reference to a separate specification (known as a ‘profile’) that defines the attributes of the actuator field. A particular actuator (or set of actuators) can be identified with increasing precision with one or more ‘actuator-specifiers’. Information with respect to how the action is to be executed is provided with one or more ‘actuator-options’.

The COMMAND OPTIONS field (optional) is populated by one or more ‘command-options’ that are identified in this specification. The command-options provide information that influences how the command is executed.

Table 2-1 summarizes the fields and subfields of an OpenC2 command. OpenC2 commands MUST contain an ACTION and TARGET and MAY contain an ACTUATOR and/or OPTION. OpenC2 is agnostic of any particular serialization; however, implementations MUST support JSON serialization of the commands.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION</td>
<td>Required. The task or activity to be performed.</td>
</tr>
<tr>
<td>TARGET</td>
<td>Required. The object of the action. The ACTION is performed on the TARGET.</td>
</tr>
<tr>
<td>type</td>
<td>Required. The specific type of TARGET.</td>
</tr>
<tr>
<td>target-specifier</td>
<td>Optional. The specifier further identifies the target to some level of precision, such as a specific target, a list of targets, or a class of targets.</td>
</tr>
<tr>
<td>target-option</td>
<td>Optional. Additional information about how to perform the action for a specific target type.</td>
</tr>
<tr>
<td>ACTUATOR</td>
<td>Optional. The ACTUATOR may perform the ACTION on the TARGET.</td>
</tr>
</tbody>
</table>
The TARGET of an OpenC2 command may include a set of targets of the same type, a range of targets, or a particular entity. Specifiers for TARGETs are optional and provide additional precision for the target.

The OpenC2 ACTUATOR field provides information about a function that will execute the ACTION on the TARGET. Specifiers for actuators provide additional information to refine the command so that a particular function, system, class of devices, or specific device can be identified. Options for actuators provide additional information to refine the command to indicate how an action is to be done in the context of the actuator. Options are distinct from COMMAND OPTIONS in that options are a function of the actuator and the action.

COMMAND OPTIONS influence the command by providing information such as time, periodicity, duration, or other details on what is to be done. They can also be used to convey the need for acknowledgement or additional status information about the execution of a command.

### 2.2.2 Action Vocabulary

This section defines the set of OpenC2 actions grouped by their general activity. Table 2-2 summarizes the definition of the OpenC2 actions.

- **Actions that Control Information:**
  These actions are used to gather information needed to determine the current state or enhance cyber situational awareness. These actions typically do not impact the state of the target.

- **Actions that Control Permissions:**
  These actions are used to control permissions and manage accesses.

- **Actions that Control Activities/Devices:**
  These actions are used to control the state or the activity of a system, a process, a connection, a host, or a device. The actions are used to execute tasks, adjust configurations, set and update parameters, and modify attributes.

- **Effects-Based Actions:**
  Effects-based actions are at a higher level of abstraction for purposes of communicating a desired impact rather than a command to execute specific tasks within an enclave. This level of abstraction enables coordinated actions between enclaves, while permitting a local enclave to optimize its workflow for its specific environment.
Effects-based action assumes that the recipient enclave has a decision making capability because effects-based actions typically do not have a one-to-one mapping to the other actions.

### Table 2-2. Summary of Action Definitions

<table>
<thead>
<tr>
<th>Actions that Control Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>scan</strong></td>
</tr>
<tr>
<td><strong>locate</strong></td>
</tr>
<tr>
<td><strong>query</strong></td>
</tr>
<tr>
<td><strong>report</strong></td>
</tr>
<tr>
<td><strong>notify</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Actions that Control Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>deny</strong></td>
</tr>
<tr>
<td><strong>contain</strong></td>
</tr>
<tr>
<td><strong>allow</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Actions that Control Activities/Devices</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>start</strong></td>
</tr>
<tr>
<td><strong>stop</strong></td>
</tr>
<tr>
<td><strong>restart</strong></td>
</tr>
<tr>
<td><strong>pause</strong></td>
</tr>
<tr>
<td><strong>resume</strong></td>
</tr>
<tr>
<td><strong>cancel</strong></td>
</tr>
<tr>
<td><strong>set</strong></td>
</tr>
<tr>
<td><strong>update</strong></td>
</tr>
<tr>
<td><strong>move</strong></td>
</tr>
<tr>
<td><strong>redirect</strong></td>
</tr>
<tr>
<td><strong>delete</strong></td>
</tr>
<tr>
<td><strong>snapshot</strong></td>
</tr>
<tr>
<td><strong>detonate</strong></td>
</tr>
<tr>
<td><strong>restore</strong></td>
</tr>
<tr>
<td><strong>save</strong></td>
</tr>
<tr>
<td><strong>throttle</strong></td>
</tr>
<tr>
<td><strong>delay</strong></td>
</tr>
</tbody>
</table>
substitute | The ‘substitute’ action replaces all or part of the data, content, or payload.
copy | The ‘copy’ action duplicates a file or data flow.
sync | The ‘sync’ action synchronizes a sensor or actuator with other system components.

<table>
<thead>
<tr>
<th>Effects-Based Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>investigate</td>
</tr>
<tr>
<td>mitigate</td>
</tr>
<tr>
<td>remediate</td>
</tr>
</tbody>
</table>

2.2.3 Target Vocabulary

The TARGET is the object of the ACTION (or alternatively, the ACTION is performed on the TARGET). The baseline set of TARGETs is summarized in Table TBD and a full description of the targets and their associated specifiers is documented in the property tables (TBSL).

Table TBD. Summary of Target Definitions.

<table>
<thead>
<tr>
<th>Target</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBSL</td>
<td>TBSL</td>
</tr>
</tbody>
</table>

2.2.4 Actuator

An ACTUATOR is an implementation of a cyber defense function that executes the ACTION on the TARGET. An actuator profile is a specification that identifies the subset of actions, targets and other aspects of this language specification that are meaningful in the context of a particular ACTUATOR. The actuator profile also identifies the portions of this specification that are mandatory to implement as well as optional actions and also defines appropriate actuator specifiers and the actuator options.

An Actuator Profile SHALL be composed in accordance with the following framework:

TBS

2.2.5 Command Option Vocabulary

COMMAND OPTIONS influence a command and are independent of the TARGET, ACTUATOR and ACTION itself. COMMAND OPTIONS provide additional information to refine how the command is to be performed such as time, periodicity, or duration, or convey the need for status information such as a response is required. The requested status/information will be carried in a RESPONSE.

TBS

2.3 OpenC2 Response

The OpenC2 Response is a message sent from an entity as the result of a command. Response messages provide acknowledgement, status, results from a query or other information as requested from the issuer of the command. Response messages are solicited and correspond to a command. The recipient
of the OpenC2 Response is typically the entity that issued the command but any recipient(s) may be designated in the modifier of the command.

2.3.1 Response Structure

TBS

3. OpenC2 PROPERTY TABLES

TBS