# Open Command and Control (OpenC2) Language Specification Version 1.0

## Committee Specification Draft 02

## 14 February 2018

### Specification URIs

**This version:**

http://docs.oasis-open.org/openc2/oc2ls/v1.0/csd02/oc2ls-v1.0-csd02.docx
http://docs.oasis-open.org/openc2/oc2ls/v1.0/csd02/oc2ls-v1.0-csd02.html
http://docs.oasis-open.org/openc2/oc2ls/v1.0/csd02/oc2ls-v1.0-csd02.pdf

**Previous version:**

http://docs.oasis-open.org/openc2/oc2ls/v1.0/csd01/oc2ls-v1.0-csd01.pdf (Authoritative)
http://docs.oasis-open.org/openc2/oc2ls/v1.0/csd01/oc2ls-v1.0-csd01.html
http://docs.oasis-open.org/openc2/oc2ls/v1.0/csd01/oc2ls-v1.0-csd01.docx

**Latest version:**

http://docs.oasis-open.org/openc2/oc2ls/v1.0/oc2ls-v1.0.pdf
http://docs.oasis-open.org/openc2/oc2ls/v1.0/oc2ls-v1.0.html
http://docs.oasis-open.org/openc2/oc2ls/v1.0/oc2ls-v1.0.docx

**Technical Committee:**

OASIS Open Command and Control (OpenC2) TC

**Chairs:**

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

**Editors:**

Jason Romano (jdroman@nsa.gov), National Security Agency
Duncan Sparrell (duncan@sfractal.com), sFractal Consulting LLC

**Additional artifacts:**

This prose specification is one component of a Work Product that also includes:

- The Authoritative version of this specification, in the Markdown language: http://docs.oasis-open.org/openc2/oc2ls/v1.0/csd02/md/oc2ls-v1.0-wd03.md.

**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 document was last revised or approved by the OASIS Open Command and Control (OpenC2) TC on the above date. The level of approval is also listed above. Check the "Latest version" location noted above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Technical Committee (TC) are listed at https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=openc2#technical.

TC members should send comments on this specification to the TC's email list. Others should send comments to the TC's public comment list, after subscribing to it by following the instructions at the "Send A Comment" button on the TC's web page at https://www.oasis-open.org/committees/openc2/.

This specification is provided under the Non-Assertion Mode of the OASIS IPR Policy, the mode chosen when the Technical Committee was established. For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the TC's web page (https://www.oasis-open.org/committees/openc2/ipr.php).

Note that any machine-readable content (Computer Language Definitions) declared Normative for this Work Product is provided in separate plain text files. In the event of a discrepancy between any such plain text file and display content in the Work Product's prose narrative document(s), the content in the separate plain text file prevails.

**Citation format:**

**[OpenC2-Lang-v1.0]**

*Open Command and Control (OpenC2) Language Specification Version 1.0*. Edited by Jason Romano and Duncan Sparrell. 14 February 2018. OASIS Committee Specification Draft 02. http://docs.oasis-open.org/openc2/oc2ls/v1.0/csd02/oc2ls-v1.0-csd02.html. Latest version: http://docs.oasis-open.org/openc2/oc2ls/v1.0/oc2ls-v1.0.html.

# Notices



All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see https://www.oasis-open.org/policies-guidelines/trademark for above guidance.

# Table of Contents

**Editor's Note**: This document is NOT complete.

The document development process is based on agile software development principles. Iterative, incremental working documents are being developed, reviewed by the Language Subcommittee, and then submitted to the Technical Committee for approval as a Committee Specification Drafts (CSD).

This is iteration 2 and the expectation is there will be 4 or 5 CSD iterations before this document is complete and ready to be submitted for approval as a Committee Specification.

Parenthetical "Editor's Notes" will be removed prior to submitting for Committee Specification. Sections that are expected to added in a later iteration (prior to 1.0) will be labeled with "TBSL" for "To Be Supplied Later", optionally with a guestimate as to which iteration it would be supplied in.

  

# 1 Introduction

The OpenC2 Language Specification defines a language used to compose messages for command and control of cyber defense systems and components.

The OpenC2 language defines two message types:

1. **Command**: An instruction from one system to another to act on the content of the command
2. **Response**: Any information captured or necessary to send back to the invoking system that requested the Command be invoked

The components of an OpenC2 Command are an action (what is to be done), a target (what is being acted upon), an optional actuator (what is performing the command), and command options, which influence how the command is to be performed. An action coupled with a target is sufficient to describe a complete OpenC2 Command. The inclusion of an actuator and/or command-options provide additional precision.

Additional detail regarding the TARGET and ACTUATOR 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).

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

## 1.1 Goal

Editor's Note - TBSL - This section will be included in a future iteration (probably iteration 5) prior to submitting for Committee Specification.

## 1.2 Purpose and Scope

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

1. the set of actions and options that may be used in OpenC2 commands
2. the set of targets, target specifiers, and target options
3. A syntax that defines the structure of commands and responses
4. an organizational scheme that describes an Actuator Profile
5. The MTI serialization of OpenC2 commands, and responses
6. the procedures for extending the language

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. The following items are beyond the scope of this specification:

1. Language extensions unique to an actuator
2. Alternate serializations of OpenC2 commands
3. The enumeration of the protocols required for transport, information assurance, sensing, analytics and other external dependencies

## 1.3 IPR Policy

This specification is provided under the Non-Assertion Mode of the OASIS IPR Policy, the mode chosen when the Technical Committee was established. For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the TC's web page (https://www.oasis-open.org/committees/openc2/ipr.php).

## 1.4 Terminology

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] and [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 1.5 Document Conventions

Editor's Note - TBSL - This section will be included in a future iteration (probably iteration 5) prior to submitting for Committee Specification.

## 1.6 Naming Conventions

RFC2119/RFC8174 key words (see section 1.4) are in all uppercase.

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.

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

## 1.7 Normative References


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

**[RFC8174]** Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, http://www.rfc-editor.org/info/rfc8174.

# 2 OpenC2 Language

## 2.1 Overview

The OpenC2 language has two distinct message types: Command and Response. The OpenC2 Command describes an action performed on a target. The OpenC2 Response is a means to provide information (such as acknowledgement, status, etc.) as a result of an OpenC2 Command.

## 2.2 OpenC2 Command

The OpenC2 Command communicates an action to be performed on a target and may include information identifying the actuator(s) that is to execute the command.

### 2.2.1 Command Structure

An OpenC2 Command has four fields: ACTION, TARGET, ACTUATOR and COMMAND-OPTIONS.

The ACTION and TARGET fields are required and are populated by one of the 'action-types' in Table 2-1 and the 'target-types' in Table 2-2. A particular target-type may be further refined by one or more 'target-specifiers' and/or 'target-options'.

The optional ACTUATOR field identifies the entity or entities that are tasked to execute the OpenC2 Command.

Information with respect to how the action is to be executed is provided with one or more 'actuator-options'.

The optional COMMAND-OPTIONS field is populated by one or more 'command-options' that provide information that influences how the command is executed.

The following list summarizes the fields and subfields of an OpenC2 Command. OpenC2 Commands MUST contain an ACTION and TARGET and MAY contain an ACTUATOR and/or COMMAND-OPTIONS. OpenC2 is agnostic of any particular serialization; however, implementations MUST support JSON serialization of the commands.

- **ACTION** (required): The task or activity to be performed.
- **TARGET** (required): The object of the action. The ACTION is performed on the target.
  - **TARGET-NAME** (required): The name of the object of the action.
  - **TARGET-SPECIFIERS** (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.
  - **TARGET-OPTIONS** (optional): Additional information about how to perform the action for a specific target type.
- **ACTUATOR** (optional): The ACTUATOR may perform the ACTION on the TARGET. The ACTUATOR type will be defined within the context of an Actuator Profile.
  - **ACTUATOR-NAME** (required): The name of the set of functions (e.g., "firewall") performed by the actuator, and the name of the profile defining commands applicable to those functions.
  - **ACTUATOR-SPECIFIERS** (optional): The specifier identifies the actuator to some level of precision, such as a specific actuator, a list of actuators, or a group of actuators.

  - **ACTUATOR-OPTIONS** (optional): The options specify how a particular ACTION is to be performed for an actuator type.
- **COMMAND-OPTIONS** (optional): Provide additional information on how the command is to be performed, such as date/time, periodicity, duration etc. COMMAND OPTIONS only influence/ impact the command and are defined independently of any ACTION, ACTUATOR or TARGET.

The TARGET of an OpenC2 Command may include a set of targets of the same type, a range of targets, or a particular target. Specifiers for TARGETs are optional and provide additional precision for the target.

The OpenC2 ACTUATOR field identifies the entity(ies) that execute the ACTION on the TARGET. Specifiers for actuators refine the command so that a particular function, system, class of devices, or specific device can be identified. Actuator-options indicate how an action is to be done in the context of the actuator.

COMMAND-OPTIONS influence the command by providing information such as time, periodicity, duration, or other details on what is to be executed. 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 2-1 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.
- *Actions that Control Access*: These actions are used to control traffic flow and file permissions (e.g., allow/deny).
- *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. 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 2-1. Summary of Action Definitions**

| Action | Description |
|---|---|
| | **Actions that Control Information** |
| scan | The scan action is the systematic examination of some aspect of the entity or its environment in order to obtain information. |
| locate | The locate action is used to find an object either physically, logically, functionally, or by organization. |

| Action | Description |
|---|---|
| query | The query action initiates a request for information. |
| report | The report action tasks an entity to provide information to a designated recipient of the information. |
| notify | The notify action is used to set an entity's alerting preferences. |
| | **Actions that Control Access** |
| deny | The deny action is used to prevent a certain event or action from completion, such as preventing a flow from reaching a destination (e.g., block) or preventing access. |
| contain | The contain action stipulates the isolation of a file, process, or entity such that it cannot modify or access assets or processes that support the business and/or operations of the enclave. |
| allow | The allow action permits the access to or execution of a target. |
| | **Actions that Control Activities/Devices** |
| start | The start action initiates a process, application, system, or some other activity. |
| stop | The stop action halts a system or ends an activity. |
| restart | The restart action conducts a stop of a system or an activity followed by a start of a system or an activity. |
| pause | The pause action ceases a system or activity while maintaining state. |
| resume | The resume action starts a system or activity from a paused state. |
| cancel | The cancel action invalidates a previously issued action. |
| set | The set action changes a value, configuration, or state of a managed entity within an IT system. |
| update | The update action instructs the component to retrieve, install, process, and operate in accordance with a software update, reconfiguration, or some other update. |
| move | The move action changes the location of a file, subnet, or process. |
| redirect | The redirect action changes the flow to a particular destination other than its original intended destination. |
| create | The create action adds a new entity of a known type (e.g., data, files, directories). |
| delete | The delete action removes an entity of a known type (e.g., data, files, flows). |
| snapshot | The snapshot action captures the state of a target at an instant in time. |
| detonate | The detonate action executes and observes the behavior of a target (e.g., file, hyperlink) in a manner that is isolated from assets that support the business or operations of the enclave. |
| restore | The restore action returns to an identical or similar known state. |
| save | The save action commits data or system state to memory. |

| Action | Description |
|---|---|
| throttle | The throttle action adjusts the rate of a process, function, or activity. |
| delay | The delay action stops or holds up an activity or data transmittal. |
| 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 an actuator with other system components. |
| | **Effects-Based Actions** |
| investigate | The investigate action tasks the recipient to aggregate and report information as it pertains to a security event or incident. |
| mitigate | The mitigate action tasks the recipient to circumvent the problem without necessarily eliminating the vulnerability or attack point. |
| remediate | The remediate action tasks the recipient to eliminate the vulnerability or attack point. |

### 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 2-2 and a full description of the targets and their associated specifiers is documented in the property tables (TBSL).

**Table 2-2. Summary of Targets.**

| Target | Description |
|---|---|
| artifact | The Artifact Object permits capturing an array of bytes (8-bits), as a base64-encoded string or linking to a file-like payload. |
| command | The Command Object represents a reference to a previously issued OpenC2 Command. |
| device | The Device Object represents the properties of a hardware or virtual device. |
| directory | The Directory Object represents the properties common to a file system directory. |
| disk | The Disk Object represents a disk drive. |
| disk_partition | The Disk Partition Object represents a single partition of a disk drive. |
| domain_name | The Domain Name represents the properties of a network domain name. |
| email_addr | The Email Address Object represents a single email address. |
| email_message | The Email Message Object represents an instance of an email message, corresponding to the internet message format described in RFC 5322 and related RFCs. |
| file | The File Object represents the properties of a file. |

| Target | Description |
|---|---|
| ipv4_addr | The IPv4 Address Object represents one or more IPv4 addresses expressed using CIDR notation. |
| ipv6_addr | The IPv6 Address Object represents one or more IPv6 addresses expressed using CIDR notation. |
| mac_addr | The MAC Address Object represents a single Media Access Control (MAC) address. |
| memory | The Memory Object represents memory objects. |
| ip_connection | The IP Connection Object represents a network connection that originates from a source and is addressed to a destination. |
| openc2 | The OpenC2 object is the summation of the actions, targets and profiles supported by the actuator. The target is used with the query action to determine an actuator's capabilities. |
| process | The Process Object represents common properties of an instance of a computer program as executed on an operating system. |
| software | The Software Object represents high-level properties associated with software, including software products. |
| url | The URL Object represents the properties of a uniform resource locator (URL). |
| user_account | The User Account Object represents an instance of any type of user account, including but not limited to operating system, device, messaging service, and social media platform accounts. |
| user_session | The User Session Object represents a user session. |
| volume | The Volume Object represents a generic drive volume. |
| windows_registry_key | The Registry Key Object represents the properties of a Windows registry key. |
| x509_certificate | The X509 Certificate Object represents the properties of an X.509 certificate, as defined by ITU recommendation X.509. |

### 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 mandatory to implement or optional in the context of a particular ACTUATOR. An Actuator Profile also defines ACTUATOR-SPECIFIERS and ACTUATOR-OPTIONS that are meaningful and possibly unique to the actuator.

An Actuator Profile SHALL be composed in accordance with the framework in section 4.

Editor's Note - TBSL - More text be included in a future iteration (probably iteration 4) prior to submitting for Committee Specification.

### 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.

Table 2-3 lists the valid command-options.

Editor's Note - TBSL - This table be included in a future iteration (probably iteration 3) prior to submitting for Committee Specification.

**Table 2-3. Summary of Command Options.**

| Command Option | Type | Description |
|---|---|---|
| TBSL | TBSL | TBSL |

## 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.

### 2.3.1 Response Structure

Editor's Note - TBSL - This section be included in a future iteration (probably iteration 3) prior to submitting for Committee Specification.

# 3 OpenC2 Property Tables

## 3.1 OpenC2 Messages

The following subsections provide the permitted values within an OpenC2 message.

### 3.1.1 OpenC2 Command

The OpenC2 Command describes an action performed on a target. It can be directive or descriptive depending on the context.

#### 3.1.1.1 Type Name: OpenC2Command

Base Type: Record

| ID | Property Name | Type | Description |
|---|---|---|---|
| 1 | action | Action | The task or activity to be performed (i.e., the 'verb'). |
| 2 | target | Target | The object of the action. The action is performed on the target. |
| 3 | actuator | Actuator | The subject of the action. The actuator executes the action on the target. |
| 4 | command-options | Command-Options | An object containing additional properties that apply to the command. |

#### 3.1.1.2 Type Name: Action

Base Type: Enumerated

| ID | Element Name | Description |
|---|---|---|
| 1 | scan | The scan action is the systematic examination of some aspect of the entity or its environment in order to obtain information. |
| 2 | locate | The locate action is used to find an object either physically, logically, functionally, or by organization. |
| 3 | query | The query action initiates a request for information. |
| 4 | report | The report action tasks an entity to provide information to a designated recipient of the information. |
| 5 | notify | The notify action is used to set an entity's alerting preferences. |
| 6 | deny | The deny action is used to prevent a certain event or action from completion, such as preventing a flow from reaching a destination (e.g., block) or preventing access. |

| ID | Element Name | Description |
|---|---|---|
| 7 | contain | The contain action stipulates the isolation of a file, process, or entity such that it cannot modify or access assets or processes that support the business and/or operations of the enclave. |
| 8 | allow | The allow action permits the access to or execution of a target. |
| 9 | start | The start action initiates a process, application, system, or some other activity. |
| 10 | stop | The stop action halts a system or ends an activity. |
| 11 | restart | The restart action conducts a stop of a system or an activity followed by a start of a system or an activity. |
| 12 | pause | The pause action ceases a system or activity while maintaining state. |
| 13 | resume | The resume action starts a system or activity from a paused state. |
| 14 | cancel | The cancel action invalidates a previously issued action. |
| 15 | set | The set action changes a value, configuration, or state of a managed entity within an IT system. |
| 16 | update | The update action instructs the component to retrieve, install, process, and operate in accordance with a software update, reconfiguration, or some other update. |
| 17 | move | The move action changes the location of a file, subnet, or process. |
| 18 | redirect | The redirect action changes the flow to a particular destination other than its original intended destination. |
| 19 | create | The create action adds a new entity of a known type (e.g., data, files, directories). |
| 20 | delete | The delete action removes an entity of a known type (e.g., data, files, flows). |
| 21 | snapshot | The snapshot action captures the state of a target at an instant in time. |
| 22 | detonate | The detonate action executes and observes the behavior of a target (e.g., file, hyperlink) in a manner that is isolated from assets that support the business or operations of the enclave. |
| 23 | restore | The restore action returns to an identical or similar known state. |
| 24 | save | The save action commits data or system state to memory. |
| 25 | throttle | The throttle action adjusts the rate of a process, function, or activity. |
| 26 | delay | The delay action stops or holds up an activity or data transmittal. |
| 27 | substitute | The substitute action replaces all or part of the data, content, or payload. |
| 28 | copy | The copy action duplicates a file or data flow. |
| 29 | sync | The sync action synchronizes an actuator with other system components. |

| ID | Element Name | Description |
|---|---|---|
| 30 | investigate | The investigate action tasks the recipient to aggregate and report information as it pertains to a security event or incident. |
| 31 | mitigate | The mitigate action tasks the recipient to circumvent the problem without necessarily eliminating the vulnerability or attack point. |
| 32 | remediate | The remediate action tasks the recipient to eliminate the vulnerability or attack point. |

#### 3.1.1.3 Type Name: Target

Base Type: Choice

| ID | Property Name | Type | Description |
|---|---|---|---|
| 1 | artifact | artifact | The Artifact Object permits capturing an array of bytes (8-bits), as a base64-encoded string or linking to a file-like payload. |
| 2 | command | command | The Command Object represents a reference to a previously issued OpenC2 Command. |
| 3 | device | device | The Device Object represents the properties of a hardware or virtual device. |
| 4 | directory | directory | The Directory Object represents the properties common to a file system directory. |
| 5 | disk | disk | The Disk Object represents a disk drive. |
| 6 | disk_partition | disk-partition | The Disk Partition Object represents a single partition of a disk drive. |
| 7 | domain_name | domain-name | The Domain Name represents the properties of a network domain name. |
| 8 | email_addr | email-addr | The Email Address Object represents a single email address. |
| 9 | email_message | email-message | The Email Message Object represents an instance of an email message, corresponding to the internet message format described in RFC 5322 and related RFCs. |
| 10 | file | file | The File Object represents the properties of a file. |
| 11 | ipv4_addr | ipv4-addr | The IPv4 Address Object represents one or more IPv4 addresses expressed using CIDR notation. |
| 12 | ipv6_addr | ipv6-addr | The IPv6 Address Object represents one or more IPv6 addresses expressed using CIDR notation. |

| ID | Property Name | Type | Description |
|---|---|---|---|
| 13 | mac_addr | mac-addr | The MAC Address Object represents a single Media Access Control (MAC) address. |
| 14 | memory | memory | The Memory Object represents memory objects. |
| 15 | ip_connection | ip-connection | The IP Connection Object represents a network connection that originates from a source and is addressed to a destination. |
| 16 | openc2 | openc2 | The OpenC2 object is the summation of the actions, targets and profiles supported by the actuator. The target is used with the query action to determine an actuator's capabilities. |
| 17 | process | process | The Process Object represents common properties of an instance of a computer program as executed on an operating system. |
| 18 | software | software | The Software Object represents high-level properties associated with software, including software products. |
| 19 | url | url | The URL Object represents the properties of a uniform resource locator (URL). |
| 20 | user_account | user-account | The User Account Object represents an instance of any type of user account, including but not limited to operating system, device, messaging service, and social media platform accounts. |
| 21 | user_session | user-session | The User Session Object represents a user session. |
| 22 | volume | volume | The Volume Object represents a generic drive volume. |
| 23 | windows_registry_key | windows-registry-key | The Registry Key Object represents the properties of a Windows registry key. |
| 24 | x509_certificate | x509-certificate | The X509 Certificate Object represents the properties of an X.509 certificate, as defined by ITU recommendation X.509. |

#### 3.1.1.4 Type Name: Actuator

Editor's Note - TBSL - This section be included in future iterations (probably iterations 3 & 4) prior to submitting for Committee Specification.

#### 3.1.1.5 Type Name: Command-Options

Editor's Note - TBSL - This section be included in future iterations (probably iterations 3 & 4) prior to submitting for Committee Specification.

### 3.1.2 OpenC2 Response

Editor's Note - TBSL - This section be included in future iterations (probably iterations 3 & 4) prior to submitting for Committee Specification.

## 3.2 Property Details

Editor's Note - TBSL - This section be included in future iterations (probably iterations 3 & 4) prior to submitting for Committee Specification.

# 4 Foundational Actuator Profile

Editor's Note - TBSL - This section be included in a future iteration (probably iteration 5) prior to submitting for Committee Specification.

# 5 Conformance

OpenC2 is a command and control language that converges (i.e., common 'point of understanding') on a common syntax, and lexicon. Conformant implementations of OpenC2:

- MUST support OpenC2 commands and responses 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 and responses that are consistent with the syntax defined in this document.

Editor's Note - TBSL - More conformance text will be included in a future iteration (probably iteration 5) prior to submitting for Committee Specification.

# Appendix A. Acknowledgments

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

**Participants**:

Editor's Note - TBSL - This section be included in the final iteration prior to submitting for Committee Specification.

# Appendix B. Revision History

| Revision | Date | Editor | Changes Made |
|---|---|---|---|
| v1.0-wd01 | 10/31/2017 | Romano, Sparrell | Initial working draft |
| v1.0-csd01 | 11/14/2017 | Romano, Sparrell | approved wd01 |
| v1.0-wd02 | 01/12/2018 | Romano, Sparrell | 01 ballot comments<br>targets |
| v1.0-wd03 | 01/31/2018 | Romano, Sparrell | Incorporated comments from TC review: editorial, action/target definition changes |

# Appendix C. Acronyms

Editor's Note - TBSL - This section be included in the final iteration prior to submitting for Committee Specification.