OpenC2 Actuator Profile for Threat Hunting Version 1.0

Committee Specification Draft 02

21 February 2024

This stage:

https://docs.oasis-open.org/openc2/ap-hunt/v1.0/csd02/ap-hunt-v1.0-csd02.md (Authoritative)
https://docs.oasis-open.org/openc2/ap-hunt/v1.0/csd02/ap-hunt-v1.0-csd02.html
https://docs.oasis-open.org/openc2/ap-hunt/v1.0/csd02/ap-hunt-v1.0-csd02.pdf

Previous stage:

https://docs.oasis-open.org/openc2/ap-hunt/v1.0/csd01/ap-hunt-v1.0-csd01.md (Authoritative)
https://docs.oasis-open.org/openc2/ap-hunt/v1.0/csd01/ap-hunt-v1.0-csd01.html
https://docs.oasis-open.org/openc2/ap-hunt/v1.0/csd01/ap-hunt-v1.0-csd01.pdf

Latest stage:

https://docs.oasis-open.org/openc2/ap-hunt/v1.0/ap-hunt-v1.0.md (Authoritative)
https://docs.oasis-open.org/openc2/ap-hunt/v1.0/ap-hunt-v1.0.html
https://docs.oasis-open.org/openc2/ap-hunt/v1.0/ap-hunt-v1.0.pdf

Technical Committee:

OASIS Open Command and Control (OpenC2) TC

Chairs:

Duncan Sparrell (duncan@sfractal.com), sFractal Consulting LLC
Michael Rosa (mjrosa@nsa.gov), National Security Agency

Editor:

David Lemire (david.lemire@hii-tsd.com), National Security Agency

Additional artifacts:

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

- JADN and JIDL schemas: https://docs.oasis-open.org/openc2/ap-hunt/v1.0/csd02/schemas/

Related work:

This specification is related to:

- *Open Command and Control (OpenC2) Language Specification Version 1.1*. Edited by Duncan Sparrell and Toby Considine. Latest stage: https://docs.oasis-open.org/openc2/oc2ls/v1.1/oc2ls-v1.1.html.

Declared JADN Namespaces

- http://docs.oasis-open.org/openc2/ns/ap/hunt/v1.0

Abstract:

This specification defines an actuator profile to automate management of cyber threat hunting activities using OpenC2. Threat hunting is the process of proactively and iteratively searching through networks and on endpoints to detect and isolate cyber observables that may indicate threats that evade existing security solutions. This actuator profile defines the OpenC2 Actions, Targets, Arguments, and Specifiers along with conformance clauses to enable the operation of OpenC2 Producers and Consumers in the context of cyber threat hunting. It covers the identification and invocation of stored hunting processes (e.g., “hunt flows”), passing of hunt parameters, and the expected type(s) and format(s) of information returned by hunting processes.

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 stage" 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://groups.oasis-open.org/communities/tc-community-home2?CommunityKey=a34c9baf-48b2-44c5-a567-018dc7d32296#technical.

TC members should send comments on this specification to the TC's email list. Others may submit comments to the TC at Technical-Committee-Comments@oasis-open.org.

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.

Key words:

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] and [RFC8174] when, and only when, they appear in all capitals, as shown here.

Citation format:

When referencing this specification the following citation format should be used:

[AP-Hunt-v1.0]

OpenC2 Actuator Profile for Threat Hunting Version 1.0. Edited by David Lemire. 21 February 2024. OASIS Committee Specification Draft 02. https://docs.oasis-open.org/openc2/ap-hunt/v1.0/csd02/ap-hunt-v1.0-csd02.html. Latest stage: https://docs.oasis-open.org/openc2/ap-hunt/v1.0/ap-hunt-v1.0.html.

Notices

Table of Contents

1 Introduction

The content in this section is non-normative, except where it is marked normative.

Note: This Actuator profile is consistent with Version 1.0 of the OpenC2 Language Specification ([OpenC2-Lang-v1.1]).

OpenC2 is a suite of specifications that enables command and control of cyber defense systems and components. OpenC2 typically uses a request-response paradigm where a Command is encoded by a Producer (managing application) and transferred to a Consumer (managed device or virtualized function) using a secure transfer protocol, and the Consumer acts on the request and responds with status and any other requested information. An overview of the concepts that underlie OpenC2 and the structure of the suite of specifications can be found in the OpenC2 *Architecture Specification* [OpenC2-Arch-v1.0].

This specification defines an Actuator profile for **Threat Hunting (TH)**. In particular, the specification comprises a set of Actions, Targets and Target Specifiers, Command Arguments, and Actuator Specifiers that integrates TH functionality with the OpenC2 Command set. Through this Command set, cyber security orchestrators may gain visibility into and provide control over TH functionality in a manner that is independent of the instance of the TH function.

All components, devices, and systems that provide TH functionality must implement the identified OpenC2 Actions, Targets, Specifiers, and Arguments as specified in the Conformance section of this specification.

Though cyber defense components, devices, systems and/or instances may implement multiple Actuator profiles, a particular OpenC2 Message may reference at most a single Actuator profile. The scope of this document is limited to TH.

The rest of the specification is organized as follows:

The remaining of Section One includes infomation about the terminology used, document conventions, and purpose of this Actuator profile specification.

Section Two (normative) binds this particular profile to the OpenC2 Language Specification. Section Two enumerates the components of the language specification that are meaningful in the context of TH and defines components that are applicable to this distinct profile. Section Two also defines the Commands (i.e., the Action/Target pairs) that are permitted in the context of TH.

Section Three (normative) presents definitive criteria for conformance so that cyber security stakeholders can be assured that their products, instances and/or integrations are compatible with OpenC2.

> **NOTE** - fill in information about annexes / appendices as they are defined.

1.1 Changes from earlier versions

1.2 Glossary

1.2.1 Definitions of terms

This section is normative.

1.2.1.1 OpenC2 Terms

- **Action**: The task or activity to be performed (e.g., 'deny').
- **Actuator**: The function performed by the Consumer that executes the Command (e.g., 'Stateless Packet Filtering').
- **Argument**: A property of a Command that provides additional information on how to perform the Command, such as date/time, periodicity, duration, etc.
- **Command**: A Message defined by an Action-Target pair that is sent from a Producer and received by a Consumer.

- **Consumer**: A managed device / application that receives Commands. Note that a single device / application can have both Consumer and Producer capabilities.
- **Message**: A content- and transport-independent set of elements conveyed between Consumers and Producers.
- **Producer**: A manager application that sends Commands.
- **Response**: A Message from a Consumer to a Producer acknowledging a Command or returning the requested resources or status to a previously received Command.
- **Specifier**: A property or field that identifies a Target or Actuator to some level of precision.
- **Target**: The object of the Action, i.e., the Action is performed on the Target (e.g., IP Address).

1.2.1.2 Threat Hunting Terms

- **Threat Hunting**: Cyber threat hunting is a proactive security search through networks, endpoints, and datasets to hunt malicious, suspicious, or risky activities that have evaded detection by existing tools.
  Source: https://www.trellix.com/en-us/security-awareness/operations/what-is-cyber-threat-hunting.html
- **Huntbook**: A hunt flow combined with its execution results in a notebook format.
  Source: https://kestrel.readthedocs.io/en/stable/language/tac.html#huntbook
- **Huntflow**: The control flow of a hunt. A hunt flow comprises a series of hunt steps, computing multiple sets of entities, and deriving new sets of entities based on previous ones. Finally, a hunt flow reveals all sets of entities that are associated with a threat.
  Source: https://kestrel.readthedocs.io/en/stable/language/tac.html#hunt-flow
- **Datasource**: Data sources, e.g., an EDR, a SIEM, a firewall, that provide raw or processed data for hunting.
  Source: https://kestrel.readthedocs.io/en/stable/installation/datasource.html
- **Hunt Arguments**: ...

1.2.2 Acronyms and abbreviations

This section is non-normative

Term	Expansion
AP	Actuator Profile
EDR	Endpoint Detection & Response
IPR	Intellectual Property Rights
JADN	JSON Abstract Data Notation
JSON	JavaScript Object Notation
OASIS	Organization for the Advancement of Structured Information Standards
RFC	Request for Comment
SCO	STIX Cyber-observable Objects
SIEM	Security Information & Event Management
STIX	Structured Threat Information eXpression
TC	Technical Committee

Term	Expansion
TH	Threat Hunting
URI	Uniform Resource Identifier

1.2.3 Document conventions

- Naming conventions
- Font colors and styles
- Typographic conventions

1.5 Overview

Cyber threat hunting is a proactive security search through networks, endpoints, and datasets to hunt malicious, suspicious, or risky activities that have evaded detection by existing tools. Various aspects of threat hunting can be manual, machine-assisted, or automated. This AP defines the use of OpenC2 to invoke machine-assisted or automated threat hunting activities and return associated results. It assumes the availability to the OpenC2 Consumer with one or more defined huntflows that can be invoked and one or more relevant data sources that can be queried, and applies the OpenC2 introspection model to enable an OpenC2 Producer to determine the data sources and huntflows available from a particular Hunt Consumer to enable it to invoke huntflows with relevant parameters.

There are multiple common approaches to cyber threat hunting. The capabilities defined by this AP are consistent with the 2019 MITRE Technical Report, *TTP-Based Hunting* [TTP-Hunting] but should also be applicable to the use of other hunting implementations and approaches.

> Research links for threat hunting background:
>
> - https://www.ibm.com/topics/threat-hunting
> - https://www.crowdstrike.com/cybersecurity-101/threat-hunting/
> - https://www.trellix.com/en-us/security-awareness/operations/what-is-cyber-threat-hunting.html

1.6 Purpose and Scope

This Actuator profile specifies the set of Actions, Targets, Specifiers, and Command Arguments that integrates the investigation capabilities of threat hunting (TH) systems with the OpenC2 Command set. Through this Command set, cyber security orchestrators may gain visibility into and provide control over TH functionality in a manner that is independent of the instance of the threat hunting solution.

All components, devices and systems that provide TH functionality will implement the OpenC2 Actions, Targets, Specifiers and Arguments identified as required in this document. Actions that are applicable, but not necessarily required, for TH will be identified as optional.

The purpose of this document is to:

- Identify the required OpenC2 Actions for Consumers with TH functionality
- Identify the required and optional Target types for each Action in the TH class of Actuators
- Identify Actuator-Specifiers and Arguments for each Action/Target pair that are applicable and/or unique to TH
- Annotate each Action/Target pair with a justification and example, and provide sample OpenC2 Commands to a TH with corresponding Responses

This TH profile:

- Does not define or implement Actions beyond those defined in Version 1.1 of the OpenC2 *Language Specification* [OpenC2-Lang-v1.1]
- Is consistent with Version 1.1 of the OpenC2 Language Specification

2 OpenC2 Language Binding for Threat Hunting

This section is normative

This section defines the set of Actions, Targets, Specifiers, and Arguments that are meaningful in the context of TH. This section also describes the appropriate format for the status and properties of a Response frame. This section is organized into three major subsections; Command Components, Response Components and Commands.

Extensions to the Language Specification are defined in accordance with [OpenC2-Lang-v1.1], Section 3.1.5, where:

1. The unique name of the threat hunting schema is `docs.oasis-open.org/openc2/v1.0/schemas/ap-hunt-0.7.jadn`.
2. The namespace identifier (nsid) referring to the threat hunting schema is: `th`.
3. The definitions of and conformance requirements for these types are contained in this document.

2.1 OpenC2 Command Components

The components of an OpenC2 Command include Actions, Targets, Actuators and associated Arguments and Specifiers. Appropriate aggregation of the components will define a Command-body that is meaningful in the context of threat hunting.

The OpenC2 *Language Specification* [OpenC2-Lang-v1.1] identifies the applicable components of an OpenC2 Command. The components of an OpenC2 Command include:

- **Action**: A subset of the Actions defined in the OpenC2 Language Specification that are meaningful in the context of threat hunting.
  - This profile SHALL NOT define Actions that are external to Version 1.1 of the OpenC2 Language Specification
  - This profile MAY augment the definition of the Actions in the context of threat hunting
  - This profile SHALL NOT define Actions in a manner that is inconsistent with version 1.1 of the OpenC2 Language Specification
- **Target**: A subset of the Targets and Target-Specifiers defined in Version 1.0 of the OpenC2 Language Specification that are meaningful in the context of threat hunting and several Targets (and associated Specifiers) that are defined in this specification
- **Arguments**: A subset of the Arguments defined in the Language Specification and a set of Arguments defined in this specification

  > **NOTE:** Per LS PR #404, when the v2 LS progresses "Actuator" should become "Profile"

- **Actuator**: A set of specifiers defined in this specification that are meaningful in the context of threat hunting

2.1.1 Actions

Table 2.1.1-1 presents the OpenC2 Actions defined in version 1.1 of the Language Specification which are meaningful in the context of threat hunting. The particular Action/Target pairs that are required or are optional are presented in Section 2.3.

Table 2.1.1-1. Actions Applicable to Threat Hunting

Type: Action (Enumerated)

ID	Name	Description
3	**query**	Initiate a request for information.
30	**investigate**	Task the recipient to aggregate and report information as it pertains to a security event or incident.

Usage:

- The **query** action is used to determine the available capabilities of the hunting Consumer, consistent with the introspection model described in the OpenC2 *Architecture Specification* [OpenC2-Arch-v1.0].
- The **investigate** action is used to initiate a specified hunting activity.

2.1.2 Targets

This threat hunting AP employs Targets defined by the OpenC2 Language Specification and Targets specific to threat hunting functionality. The particular Action/Target pairs that are required or are optional are presented in Section 2.3.

2.1.2.1 Common Targets

Table 2.1.2-1 lists the Targets defined in the OpenC2 Language Specification that are applicable to threat hunting.

Table 2.1.2-1. Targets Applicable to Threat Hunting

Type: Target (Choice)

ID	Name	Type	#	Description
9	**features**	ls:Features	1	A set of items such as Action/Target pairs, profiles versions, options that are supported by the Actuator. The **features** Target is used with the query Action to determine an Actuator's capabilities.
1036	**th**	TH-Target	1	Hunts, Huntflows, Data sources.

2.1.2.2 Threat Hunting Targets

The list of common Targets is extended to include the additional Targets defined in this section and referenced with the `th` namespace.

Table 2.1.2-2. Targets Unique to Threat Hunting

Type: TH-Target (Choice)

ID	Name	Type	#	Description
1	**hunt**	String	1	A procedure to find a set of entities in the monitored environment that associates with a cyberthreat.
2	**huntflows**	Huntflow-Specifiers	1	The control flow of a hunt.
3	**datasources**	String	1	An identifiable collection of raw or processed data.

2.1.3 Type Definitions

Common data types are defined in [OpenC2-Lang-V1.1]. This section defines data types associated with TH activities.

Hunt arguments are used to specify the parameters of a particular hunt. This AP supports a flexible set of arguments to enable an OpenC2 Producer to access the full capabilities of an OpenC2 Hunt Consumer. Hunt arguments can take a variety of forms, including hunt-specific forms, standard OpenC2 data types and STIX objects.

Table 2.1.3-1 Hunt Arg Types

Type: TH-Args (Map)

ID	Name	Type	#	Description
1	**huntargs**	Huntargs	1	Arguments for use in conjunction with hunt implementation.

Table 2.1.3-2 Hunt Huntargs Type

Type: Huntargs (Record{1..*})

ID	Name	Type	#	Description
1	**string_args**	ArrayOf(String)	1	string arguments supplied as huntargs.
2	**integer_args**	ArrayOf(Integer)	1	integer arguments supplied as huntargs.
3	**typed_args**	Typed-Arguments	1	Paired strings of named arguments.
4	**native_oc2**	OC2-Data	1	OC2 Language types supplied as huntargs.
5	**stix**	sco:STIX-Cybersecurity-Observables	1	STIX arguments supplied as huntargs.
6	**stix_extensions**	oca:OCA-Extensions	1	STIX arguments extended with OCA extensions supplied as huntargs. TODO add a custom stix for oca-asset and event
7	**timeranges**	Timeranges	1	Timeranges used in the execution of a hunt.
8	**datasources**	Datasource-Array	1	Available data sources for hunting. These may be a host monitor, an EDR, a SIEM, a firewall, etc.

Table 2.1.3-3 OC2-Data Type

Type Name	Type Definition	Description
OC2-Data	ArrayOf(Language-Spec-Types){1..*}	OC2-Data is an array of one or more types defined in the OpenC2 language spec

Table 2.1.3-4 Language Specification Types

Type: Language-Spec-Types (Record)

ID	Name	Type	#	Description
1	**artifact**	ls:Artifact	1	An array of bytes representing a file-like object or a link to that object.
2	**device**	ls:Device	1	The properties of a hardware device.
3	**domain_name**	ls:Domain-Name	1	A network domain name.
4	**email-address**	ls:Email-Addr	1	A single email address
5	**file**	ls:File	1	Properties of a file.
6	**hashes**	ls:Hashes	1	Not used as an entity; use inside File or other attribute of another type. May be used as a query value.
7	**hostname**	ls:Hostname	1	Value must be a hostname as defined in [RFC1034], Section 3.1
8	**idn_domain_name**	ls:IDN-Domain-Name	1	An internationalized domain name.

ID	Name	Type	#	Description
9	**idn_email_address**	ls:IDN-Email-Addr	1	A single internationalized email address.
10	**ipv4_address**	ls:IPv4-Addr	1	IPv4 address as defined in [RFC0791].
11	**ipv6_address**	ls:IPv6-Addr	1	IPv6 address as defined in [RFC8200].
12	**ipv4_network**	ls:IPv4-Net	1	IPv4 network targeted by hunt activity.
13	**ipv6_network**	ls:IPv6-Net	1	IPv6 network targeted by hunt activity.
14	**ipv4_connection**	ls:IPv4-Connection	1	A 5-tuple of source and destination IPv4 address ranges, source and destination ports, and protocol.
15	**ipv6_connection**	ls:IPv6-Connection	1	A 5-tuple of source and destination IPv6 address ranges, source and destination ports, and protocol.
16	**iri**	ls:IRI	1	An internationalized resource identifier (IRI).
17	**mac_address**	ls:MAC-Addr	1	A Media Access Control (MAC) address - EUI-48 or EUI-64 as defined in [EUI].
18	**port**	ls:Port	1	Transport Protocol Port Number, [RFC6335]
19	**process**	ls:Process	1	Common properties of an instance of a computer program as executed on an operating system.
20	**uri**	ls:URI	1	A uniform resource identifier (URI).

Given the range of activities that could be sought when hunting, the introspection model is of particular value in determining what capabilities a Hunt Consumer make available to a Producer. One approach to introspection is identifying huntflows that can be invoked with specific argument types or argument names. Some introspection-oriented data types are also used as arguments when initiating a hunt.

Table 2.1.3-5 Specified Arguments Types Type

Type Name	Type Definition	Description
Specified-Arg-Types	ArrayOf(Arg-Type)	Return huntflows that take these argument types.

Table 2.1.3-6 Specified Arguments Names Type

Type Name	Type Definition	Description
Specified-Arg-Names	ArrayOf(Arg-Name)	Return huntflows that take arguments with these names.

The Typed-Arguments type provides a name-value structure useful for querying about available huntflows and providing arguments to a huntflow when it is invoked.

Table 2.1.3-7 Typed-Arguments Type

Type Name	Type Definition	Description
Typed-Arguments	MapOf(Arg-Name, Arg-Type)	Argument names and types tied to a specific Huntflow.

Table 2.1.3-8 Arg-Type Type

Type Name	Type Definition	Description
Arg-Type	String	Argument types used by a Huntflow. Follow STIX naming conventions, with lowercase characters and hyphens replacing spaces. Common types include process, file, and network-traffic.

Table 2.1.3-9 Arg-Name Type

Type Name	Type Definition	Description
Arg-Name	String	Argument names used by a Huntflow. Follow C variable naming conventions. Examples include name, src_port, and x_unique_id.

When executing a hunt an important parameter is the time period to be examined. Time ranges are used to specify the time period over which the hunt invoked with an `investigate /hunt` command should examine data.

Table 2.1.3-10 Timeranges Type

Type Name	Type Definition	Description
Timeranges	ArrayOf(Timerange)	a timerange used in the execution of a hunt.

Time ranges may be be specified in absolute terms, with a specific start and end time, or for a relative duration leading up to the present time.

Table 2.1.3-11 Timerange Type

Type: Timerange (Choice)

ID	Name	Type	#	Description
1	**timerange_absolute**	Timerange-Abs	1	Absolute timerange, defined by a start and end time in ISO 8601 format.
2	**timerange_relative**	Timerange-Rel	1	Relative timerange, example '3, Days' for last 3 days.

An absolute timerange is specified using exact start and stop times.

Table 2.1.3-12 Absolute Timerange Type

Type: Timerange-Abs (Record{2..*})

ID	Name	Type	#	Description
1	**hunt_start_time**	timestamp	1	Start time, as a STIX time string.
2	**hunt_stop_time**	timestamp	1	Stop time, as a STIX time string.

Table 2.1.3-13 timestamp Type

Type Name	Type Definition	Description
timestamp	String (%^[0-9]{4}-[0-9]{2}-[0-9]{2}T[0-9]{2}:[0-9]{2}:[0-9]{2}Z$%)	

Relative timeranges can be specified in units ranging from seconds to days. A relative timerange is always relative to the current time, so should be interpreted as specifying "the last *t* Time-units".

Table 2.1.3-14 Time-Unit Type

Type: Time-Unit (Enumerated)

ID	Name	Description
1	**Days**	
2	**Hours**	
3	**Minutes**	
4	**Seconds**	

Table 2.1.3-15 Timerange-Rel Type

Type: Timerange-Rel (Record{2..*})

ID	Name	Type	#	Description
1	**number**	Integer{0..*}	1	Number of specified Time Units used in Relative Timerange.
2	**time_unit**	Time-Unit	1	Time Unit Keywords.

2.1.4 Command Arguments

The list of common Command Arguments is extended to include the additional Command Arguments defined in this section and referenced with the `th` namespace.

Table 2.1.4-1. Command Arguments Unique to Theat Hunting

Standard OpenC2 Language arguments are available for using in threat hunting commands.

Type: Args (Map{1..*})

ID	Name	Type	#	Description
1	**start_time**	ls:Date-Time	1	
2	**stop_time**	ls:Date-Time	1	
3	**duration**	ls:Duration	1	
4	**response_requested**	ls:Response-Requested	1	
1036	**th**	TH-Args	1	

2.1.5 Actuator Specifiers

The `th` actuator specifiers provide a query toolkit for a Producer to identify the huntflows available on a Hunt Consumer and determine how to correctly invoke them. Huntflows Specifiers. They enable querying for huntflows:

- In a specified location
- By one or more descriptive labels
- By particular argument names or types

The specifiers also provide the capability to identify specific information types about the huntflow to be returned in the query response.

Table 2.1.5-1 AP Hunt Actuator Type

Type: Actuator (Enumerated)

ID	Name	Description
1036	**th**	

Table 2.1.5-2 TH Huntflow Specifiers Type

Type: Huntflow-Specifiers (Map)

ID	Name	Type	#	Description
1	**path**	String	1	Return huntflows at and below this filesystem location (absolute path).
2	**tags**	Tags	1	Return huntflows with these keywords.
3	**arg_types**	Specified-Arg-Types	1	Return huntflows that take these argument types.
4	**arg_names**	Specified-Arg-Names	1	Return huntflows that take these argument types.
5	**format_types**	Return-Type	1	Return huntflows that produce these output types.
6	**return_format**	Huntflow-Sections	1	For each huntflow returned, include these data items.

Table 2.1.5-3 Tags Type

Tags provide a mechanism to associated multiple, discrete labels to a huntflow for the purposes of organization and lookup.

Type Name	Type Definition	Description
Tags	ArrayOf(String)	Tags applied for search or filter purposes.

Table 2.1.5-4 Return-Type Type

Type: Return-Type (Record{2..*})

ID	Name	Type	#	Description
1	**var_name**	Arg-Name	1	Variable name to be returned by use of Huntflow.
2	**var_type**	Arg-Type	1	Type of data to be returned by use of Huntflow.

Table 2.1.5-5 Huntflow Sections Type

The Huntflow-Sections / Huntflow-Section types are used to specify the types of desired return information when querying a Hunt Consumer about available huntflows.

Type Name	Type Definition	Description
Huntflow-Sections	ArrayOf(Huntflow-Section)	For each huntflow returned, include these data items.

Table 2.1.5-6 Huntflow Section Type

Type: Huntflow-Section (Enumerated)

ID	Name	Description
1	**path**	Specifies the return should include the path to each Huntflow specified by the query conditions.
2	**uniqueId**	Specifies the return should include the ID of each Huntflow specified by the query conditions.
3	**version**	Specifies the return should include the version of each Huntflow specified by the query conditions.
4	**args_required**	Specifies the returned data should include the required arguments for the available Huntflows.
5	**expected_returns**	Specifies the returned data should include the expected returns for the available Huntflows.
6	**script**	Specifies the returned data should include the full text of the Huntflow for each available Huntflow.

2.2 OpenC2 Response Components

The Hunt AP supports the standard OpenC2 introspection response types plus additional hunting-oriented types.

Table 2.2-1 Results Type

Type: Results (Map{1..*})

ID	Name	Type	#	Description
1	**versions**	ls:Version	1	List of OpenC2 language versions supported by this Actuator
2	**profiles**	ArrayOf(ls:Nsid)	1	List of profiles supported by this Actuator
3	**pairs**	Pairs	1	List of targets applicable to each supported Action
4	**rate_limit**	Number{0..*}	1	
1036	**th**	TH-Results	1	

Hunting-oriented returns enable the Producer to assess the huntflows and data sources available from the Hunt Consumer, as well as return t the results of hunts invoked via the **investigate** Action.

Table 2.2-2 Threat Hunting Reponse Components

Type: TH-Results (Map{1..*})

ID	Name	Type	#	Description
1	**huntflow_info**	Huntflow-Array	1	Structured data returned by Query: Huntflows.
2	**datasources**	Datasource-Array	1	Datasource names and info returned by Query Datasources.
3	**inv_returns**	Inv-Returns	1	STIX SCO object returns.

The huntflows available from a Hunt Consumer are described using an array of Huntflow-Info records, each of which provides a collection of information identifying and characterizing an individual huntflow.

Table 2.2-3 Threat Hunting Reponse Type: Huntflow-Array

Type Name	Type Definition	Description
Huntflow-Array	ArrayOf(Huntflow-Info)	Structured data returned by Query: Huntflows.

Table 2.2-4 Threat Hunting Reponse Type: Huntflow-Info Info

Type: Huntflow-Info (Record{1..*})

ID	Name	Type	#	Description
1	**path**	String	1	Path used to identify a Huntflow in place of a name.
2	**uniqueId**	Integer{0..*}	1	Unique ID associated with a specified Huntflow.
3	**version**	String	1	Unique ID associated with a specified Huntflow.
4	**args_required**	Typed-Arguments	1	List of arguments used in the specified Huntflow.
5	**expected_returns**	Typed-Arguments	1	Data returned by the specified Huntflows.
6	**script**	String	1	Text of Hunt logic implemented by specified Huntflow.

The datasources available from a Hunt Consumer are described using an array of Datasource records, each of which provides a name and collection of descritive tags for an individual data source.

Table 2.2-5 Threat Hunting Reponse Type: Datasource Array

Type Name	Type Definition	Description
Datasource-Array	ArrayOf(Datasource)	An Array of Datasources, with multiple uses in Threathunting

Table 2.2-6 Threat Hunting Reponse Type: Datasource Type

Type: Datasource (Record{1..*})

ID	Name	Type	#	Description
1	**ds_name**	String	1	Name of a Datasource used by a Huntflow in Kestrel runtime.
2	**ds_tags**	Tags	1	Tags applied to a Datasource for search or filter purposes.

The results of a hunt are returned from a Hunt Consumer in an array of Inv[estigation]-Return records, each of which may contain one or more results represented as Strings or organized as STIX Cyber-security Observable Objects (SCOs).

Table 2.2-7 Threat Hunting Reponse Type: Inv-Returns

Type Name	Type Definition	Description
Inv-Returns	ArrayOf(Inv-Return){1..*}	Array of returns from threat hunt investigations.

Table 2.2-8 Threat Hunting Reponse Type: Inv-Return

Type: Inv-Return (Record)

ID	Name	Type	#	Description

ID	Name	Type	#	Description
1	**string_returns**	ArrayOf(String)	1	String return from an investigation
2	**stix_sco**	sco:STIX-Cybersecurity-Observables	1	STIX SCO object returns

2.2.1 Response Status Codes

2.3 OpenC2 Commands

An OpenC2 Command consists of an Action/Target pair and associated Specifiers and Arguments. This section enumerates the allowed Commands and presents the associated Responses.

Table 2.3-1 defines the Commands that are valid in the context of the threat huntung profile. An Action (the top row in Table 2.3-1) paired with a Target (the first column in Table 2.3-1) defines a valid Command. The subsequent subsections provide the property tables applicable to each OpenC2 Command.

Table 2.3-1 Command Matrix

	query	investigate
features	valid	
/huntflows	valid	
/datasources	valid	
/hunt		valid

Table 2.3-2 defines the Command Arguments that are valid for each of the commands defines in the threat huntung profile. A Command (the top row in Table 2.3-2) paired with an Argument (the first column in Table 2.3-2) defines an allowable combination. The subsection identified at the intersection of the Command/Argument provides details applicable to each Command as influenced by the Argument.

Table 2.3-2 Command Arguments Matrix

		query features	query /huntflows	query /datasources	investigate /hunt	
	response_requested	2.3.1.1	2.3.1.2	2.3.1.3	2.3.2	
	other argument #1					
	other argument #2					
	...					
	other argument *n*					

2.3.1 Query

2.3.1.1 Query Features

The `query features` Command MUST be implemented in accordance with Version 1.1 of the [OpenC2-Lang-v1.1].

2.3.1.2 Query /huntflows

The `query /huntflows` command is used to identify the set of huntflows available from a specific threat hunting consumer.

OpenC2 Consumers that receive a `query /huntflows` Command:

- but cannot parse or process the Command
  - MUST NOT respond with a OK/200
  - SHOULD respond with status code 400
  - MAY respond with the 500 status code
- but do not support the `/huntflows` Target
  - MUST NOT respond with a OK/200
  - SHOULD respond with status code 501
  - SHOULD respond with "Command not supported" in the status text
  - MAY respond with status code 500

Example `query / huntflows` commands:

- Query huntflows by argument type (link to example in appendix)
- Query huntflows by location (e.g., in a hierarchically organized collection of huntflows) (link to example in appendix)

2.3.1.3 Query /datasources

The `query /datasources` command is used to identify the set of data sources available from a specific threat hunting consumer.

OpenC2 Consumers that receive a `query /datasources` Command:

- but cannot parse or process the Command
  - MUST NOT respond with a OK/200
  - SHOULD respond with status code 400
  - MAY respond with the 500 status code
- but do not support the `/datasources` Target
  - MUST NOT respond with a OK/200
  - SHOULD respond with status code 501
  - SHOULD respond with "Command not supported" in the status text
  - MAY respond with status code 500

Example `query / datasources` commands:

- Query datasources by name (link to example in appendix)
- Query datasources by tag (e.g., for the types of techniques or known adversaries they are intended to detect) (Example 3.4)

2.3.2 Investigate /hunt

The `investigate /hunt` command is used to instigate the use of a selected huntflow in combination with a specified set of threat hunting arguments.

OpenC2 Producers that send `investigate /hunt` Commands:

- MAY populate the Command Arguments field with *fill in with appropriate TH arguments*

OpenC2 Consumers that receive a `investigate /hunt` Command:

- but cannot parse or process the Command
  - MUST NOT respond with a OK/200
  - SHOULD respond with status code 400
  - MAY respond with the 500 status code
- but do not support the `/hunt` Target
  - MUST NOT respond with a OK/200

  - SHOULD respond with status code 501
  - SHOULD respond with "Command not supported" in the status text
  - MAY respond with status code 500

Example `investigate /hunt` commands:

- Investigate ?? by absolute time (link to example in appendix)
- Investigate ?? by relative time (link to example in appendix)

3 Conformance

This section is normative.

3.1 Conformance Targets

This AP defines two conformance targets:

- **TH Producer** -- an OpenC2 Producer that creates and transmits requests consistent with this AP
- **TH Consumer** -- an OpenC2 Consumer that receives and processes requests consistent with this AP and returns corresponding responses

3.2 Conformance Requirements

3.2.1 General TH Conformance Requirements

All TH Producers and Consumers MUST:

1. Conform to the requirements of the OpenC2 Architecture Specification
2. Conform to the requirements of the OpenC2 Language Specification

3.2.2 TH Producer Conformance Requirements

TH Producers MUST:

1. Generate and transmit the `"query features"` command as defined in Section 4.1 of the OpenC2 Language Specification.
2. Generate and transmit the `"query /huntflows"` command as defined in this specification and process corresponding responses.
3. Generate and transmit the `"investigate /hunt"` command as defined in this specification and process corresponding responses.

TH Producers SHOULD:

1. Conform with at least one OpenC2 transfer specification.
2. Generate and transmit the `"query /datasources"` command as defined in this specification and process corresponding responses.

3.2.3 TH Consumer Conformance Requirements

TH Consumers MUST:

1. Receive and process the `"query features"` command as defined in Section 4.1 of the OpenC2 Language Specification and return corresponding responses.
2. Receive and process the `"query /huntflows"` command as defined in this specification and return corresponding responses.
3. Receive and process the `"investigate /hunt"` command as defined in this specification and return corresponding responses.

TH Producers SHOULD:

1. Conform with at least one OpenC2 transfer specification.
2. Receive and process the `"query /datasources"` command as defined in this specification and return corresponding responses.

Annex A. Schemas

> **NOTE:** This will become a standard section of OpenC2 APs to align with the ITU-T convention that an Annex is part of the normative content, whereas an Appendix is not.

This AP specification is composed of:

- This specification document
- The JADN schema for the TH AP, in the separate file `ap-hunt-0.7.jadn`
- The JADN schema for the TH AP in JDIL format, in the separate file `ap-hunt-0.7.jidl`

In the event of any conflict among these represenations, the contents of `ap-hunt-0.7.jadn` SHALL be considered authoritative.

Appendix A. References

This appendix contains the normative and informative references that are used in this document.

While any hyperlinks included in this appendix were valid at the time of publication, OASIS cannot guarantee their long-term validity.

A.1 Normative References

The following documents are referenced in such a way that some or all of their content constitutes requirements of this document.

(Reference sources: For references to IETF RFCs, use the approved citation formats at:
http://docs.oasis-open.org/templates/ietf-rfc-list/ietf-rfc-list.html.
For references to W3C Recommendations, use the approved citation formats at:
http://docs.oasis-open.org/templates/w3c-recommendations-list/w3c-recommendations-list.html.
Remove this note before submitting for publication.)

[OpenC2-Arch-v1.0]

Open Command and Control (OpenC2) Architecture Specification Version 1.0. Edited by Duncan Sparrell. 30 September 2022. OASIS Committee Specification 01. https://docs.oasis-open.org/openc2/oc2arch/v1.0/cs01/oc2arch-v1.0-cs01.html. Latest stage: https://docs.oasis-open.org/openc2/oc2arch/v1.0/oc2arch-v1.0.html.

[OpenC2-Lang-v1.1]

Open Command and Control (OpenC2) Language Specification Version 1.1. Edited by Duncan Sparrell and Toby Considine. Latest stage: https://docs.oasis-open.org/openc2/oc2ls/v1.1/oc2ls-v1.1.html

[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.

A.2 Informative References

[RFC3552]

Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, https://www.rfc-editor.org/info/rfc3552.

[TTP-Hunting]

Daszczyszak, R., Ellis, D., Luke, S., and Whitley, S., "TTP-Based Hunting", The MITRE Corporation, March 2019, https://www.mitre.org/sites/default/files/2021-11/prs-19-3892-ttp-based-hunting.pdf

Appendix B. Safety, Security and Privacy Considerations

(Note: OASIS strongly recommends that Technical Committees consider issues that might affect safety, security, privacy, and/or data protection in implementations of their specification and document them for implementers and adopters. For some purposes, you may find it required, e.g. if you apply for IANA registration.

While it may not be immediately obvious how your specification might make systems vulnerable to attack, most specifications, because they involve communications between systems, message formats, or system settings, open potential channels for exploit. For example, IETF [RFC3552] lists “eavesdropping, replay, message insertion, deletion, modification, and man-in-the-middle” as well as potential denial of service attacks as threats that must be considered and, if appropriate, addressed in IETF RFCs.

In addition to considering and describing foreseeable risks, this section should include guidance on how implementers and adopters can protect against these risks.

We encourage editors and TC members concerned with this subject to read *Guidelines for Writing RFC Text on Security Considerations*, IETF [RFC3552], for more information.

Remove this note before submitting for publication.)

Appendix C. Acknowledgments

Note: A Work Product approved by the TC must include a list of people who participated in the development of the Work Product. This is generally done by collecting the list of names in this appendix. This list shall be initially compiled by the Chair, and any Member of the TC may add or remove their names from the list by request. Remove this note before submitting for publication.

C.1 Special Thanks

Substantial contributions to this document from the following individuals are gratefully acknowledged:

Participant Name, Affiliation or "Individual Member"

C.2 Participants

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

OpenC2 TC Members:

First Name	Last Name	Company
Philippe	Alman	Something Networks
Alex	Amirnovman	Company B
Kris	Anderman	Mini Micro
Darren	Anstman	Big Networks

Appendix D. Revision History

Revision	Date	Editor	Changes Made
ap-hunt-v1.0-wd01	yyyy-mm-dd	Editor Name	Initial working draft

Appendix E. Threat Hunting Command / Response Examples

E.1 Example 1: Query Features

```
  "action": "query",
  "target": {
    "features": [
      "pairs"
    ]
  }
}
```

A Language Specification command, Query: Features is used to gather information from consumers about their OpenC2 capabilities. A Response may

```
  "results": {
    "pairs": [
      "query: features, /huntflows, /datasources",
      "investigate: /hunt"
    ]
  },
  "status": "OK"
}
```

E.2 Example 2: Query huntflows

```
    "action": "query",
    "target": {
        "th": {
            "huntflows": {
                "tags": "searchable_tag",
                "format_types": {
                    "var_name": "desired_return_variable"
                }
            }
        }
    }
}
```

Query is extended in this profile to include additional targets. huntflows and Datasources are available as Targets to provide data to gather information about Threat Hunting processes. This command makes use of the "tags" and "format_types" specifiers (with example values) to filter the list of threathunting processes are listed as available from the consumer. This example command also makes use of the optional command_id field, that is not required to be sent in every command, but is supported in the OpenC2 Language Specification.

E.3 Example 3: Investigate Hunt

```
{
    "action": "investigate",
    "target": {
        "th": {
            "hunt": {
                "path_relative": "path/name/example"
            }
        }
    },
  "args": {
    "response_requested": "status",
    "th": {
      "huntargs": {
        "timerange": {
          "timerange_relative": {
            "number": "15",
            "time_unit": "Minutes"
          }
        },
        "datasource": "Datasource_Name",
        "hunt_process": {
          "uuid": "1234567890"
        }
      }
    }
  }
}
```

E.4 Example 4: Query Datasources by Tags

```
{
 "action": "query",
 "target": {
  "th": {
   "datasources": ""
  }
 },
 "args": {
  "th": {
   "huntargs": {
    "datasources": [{
      "ds_tags": ["apt47", "ttp61"]
     }]
   }
  }
 }
}
```

```
{
 "status": "200",
 "results": {
  "th": {
   "datasources": [{
     "ds_name": "source_1",
     "ds_tags": ["apt47", "apt54", "ttp61", "netflow_data"]
    }]
  }
 }
}
```

Appendix F. Notices

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.

As stated in the OASIS IPR Policy, the following three paragraphs in brackets apply to OASIS Standards Final Deliverable documents (Committee Specification, Candidate OASIS Standard, OASIS Standard, or Approved Errata).

[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 Standards Final Deliverable, 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 deliverable.]

[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 OASIS Standards Final Deliverable 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 OASIS Standards Final Deliverable. 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 OASIS Standards Final Deliverable 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 Standards Final Deliverable, 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.