Open Command and Control (OpenC2) Profile for Stateless Packet Filtering Firewall Functions Version 1.0

Committee Specification Draft 01

31 May 2018

Specification URIs

This version:

http://docs.oasis-open.org/openc2/oc2slpf/v1.0/csd01/oc2slpf-v1.0-csd01.html
http://docs.oasis-open.org/openc2/oc2slpf/v1.0/csd01/oc2slpf-v1.0-csd01.pdf

Previous version:

N/A

Latest version:

http://docs.oasis-open.org/openc2/oc2slpf/v1.0/oc2slpf-v1.0.html
http://docs.oasis-open.org/openc2/oc2slpf/v1.0/oc2slpf-v1.0.pdf

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:

Joe Brule (jmbrule@nsa.gov), National Security Agency
Duncan Sparrell (duncan@sfractal.com), sFractal Consulting LLC
Alex Everett (alex.everett@unc.edu), University of North Carolina, Chapel Hill

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/oc2slpf/v1.0/csd01/md/oc2slpf-v1.0-wd01.md.

Abstract:

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. Stateless packet filtering is a cyber defense mechanism that denies or allows traffic based on static properties of the traffic (such as address, port, protocol etc). This profile defines the actions, targets, specifiers and options that are consistent with version 1.0 of the OpenC2 Language Specification in the context of stateless packet filtering.

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:

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

[OpenC2-SLPF-v1.0]

Open Command and Control (OpenC2) Profile for Stateless Packet Filtering Firewall Functions Version 1.0. Edited by Joe Brule, Duncan Sparrell, and Alex Everett. 31 May 2018. OASIS Committee Specification Draft 01. http://docs.oasis-open.org/openc2/oc2slpf/v1.0/csd01/oc2slpf-v1.0-csd01.html. Latest version: http://docs.oasis-open.org/openc2/oc2slpf/v1.0/oc2slpf-v1.0.html.

Notices

Table of Contents

1 Introduction

A firewall is a policy enforcement mechanism that restricts or permits traffic based on some combination of attributes such as connection state, ports, protocols, patterns, flows etc. A 'Stateless-Packet-Filter' (SLPF) bases its policy on static values such as source address, destination address, and/or port numbers. A Stateless-Packet-Filter does not consider traffic patterns, connection state, data flows, applications, or payload information. The scope of this profile is limited to Stateless-Packet-Filtering firewalls herein referred to as SLPF firewalls. This actuator profile specifies the set of actions, targets, specifiers, and command arguments that integrates SLPF firewall functionality with the Open Command and Control (OpenC2) command set. Through this command set, cyber security orchestrators may gain visibility and provide control into the firewall functionality in a manner that is independent of the vendor or generator of the firewall.

1.1 Purpose and Scope

All components, devices and systems that provide SLPF firewall functionality will implement the ACTIONS, TARGETS, SPECIFIERS and ARGUMENTS identified as required in this document. Actions that are applicable, but not necessarily required for SLPF firewalls will be identified as optional.
The purpose of this document is to:

- Identify the required and optional OpenC2 ACTIONS for actuators with SLPF firewall functionality.
- Identify the required and optional TARGET types and associated specifiers for each action in the SLPF firewall class of actuators.
- Identify ACTUATOR SPECIFIERS, ACTUATOR-ARGUMENTS and COMMAND-ARGUMENTS for each action-target pair that are applicable and/or unique to the SLPF firewall class of actuators
- Annotate each Action/ Target pair with a justification and example and provide sample OpenC2 commands to a SLPF firewall with corresponding responses
- Provide an abstract schema that captures the specifiers and options for a SLPF firewall

This SLPF firewall profile:

- Does not define or implement ACTIONS beyond those defined in Version 1.0 of the Language Specification.
- Is conformant with version 1.0 of the OpenC2 Language Specification

Cyber defense systems that are utilizing OpenC2 may require the following components to implement the SLPF firewall profile:

- OpenC2 Producers: Devices that send commands, receive responses, and manage the execution of commands involving one or more SLPF firewalls or other actuators with SLPF firewall capability. The OpenC2 producer needs a priori knowledge of which commands the actuator

can process and execute, therefore must understand the profiles for any device that it intends to command.

- Devices that receive OpenC2 commands and implement the stateless packet-filtering firewall profile. Generally these are actuators (i.e., the device implementing the filter) but could be orchestrators (i.e., device that forwards command to the device(s) implementing the filter).
- An Openc2 Consumer may provide multiple cyber defense mechanisms including stateless packet-filtering firewall functionality as a subset of its capabilities, thus must implement the SLPF firewall profile.

1.2 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].

1.3 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, OPTIONS.
'single quotes'
Used for single actions or commands (i.e. action target pairs)

1.4 Document Overview

TBD

1.5 Normative References

TBD

1.6 Non normative References

TBD

1.7 Acknowledgements

TBD

2 OpenC2 Language Binding

This section defines the set of ACTIONS, TARGETS, SPECIFIERS and ARGUMENTS that are meaningful in the context of SLPF firewalls. This section also describes the format of the RESPONSE frame's status and results field. This section organized into three major subsections; Components, Response and Commands.
An OpenC2 command consists of an ACTION/TARGET pair and associated specifiers and arguments. This section will enumerate the allowed commands, identify which are required and present the associated responses.

2.1 OpenC2 Components

The components of an OpenC2 command include ACTIONS, TARGETS, ACTUATOR and associated ARGUMENTS and SPECIFIERS which were defined independently. Appropriate aggregation of the components will define a command-body that is meaningful in the context of a firewall.
The components of an OpenC2 command include:

- ACTIONS: A subset of the ACTIONS defined in the OpenC2 Language specification.

- This profiles does not define ACTIONS that are external to Version 1.0 of the OpenC2 Language Specification.
- This section MAY augment the definition of the actions in the context of a firewall but does not define the actions in a manner that is inconsistent with version 1.0 of the OpenC2 language specification .

- TARGETS: A subset of the TARGETs and target-specifiers defined in the Language specification that are meaningful in the context of firewalls or a TARGET and target-specifiers that was defined in this specification.
- ARGUMENTS: A subset of the COMMAND-ARGUMENTS defined in the Language Specification.
- ACTUATOR: A set of specifiers and ACTUATOR-ARGUMENTS that are defined in this specification that are meaningful in the context of SLPF firewalls.

2.1.1 Actions

Table 1 presents the OpenC2 actions that are meaningful in the context of an SLPF firewall and identifies which actions are required for at least one target type. The particular action/ target pairs that are required or optional are presented in section XXX.

Table 1. SLPF Firewall Actions

Action	Description	Req/ Opt
query	The query action initiates a single request for information. Used to communicate the supported options and determine the state or settings of the firewall.	Required
deny	The deny action is used to prevents traffic or access	Required

allow	The allow action permits traffic or access.	Required
delete	Remove firewall rule(s) as defined in section 2.1.4.2	Optional

2.1.2 Targets

Table 2 lists the TARGETs that are applicable to firewalls and identifies the which target types are required for at least one action/target pair.

Table 2. Target Data Model Applicable to SLPF Firewalls

Target	Description/Notes	Req/Opt
ip_connection	Consists of the address (source and destination), port number (source and destination) and protocol identifier. An 'incomplete' five-tuple may be sent to the firewall. Default for unspecified is “all”.	Required
openc2	Used to determine the profile(s) and imported targets supported by the actuator	Required
ap-slpf	Used to enable an openc2 producer to determine the current state or settings of the firewall. This is an optional target type defined in this specification.	Optional
ip_addr	In the context of an SLPF firewall, identifies IP address(es) that are to be denied (or allowed) regardless of source or destination. All other aspects of the ip-connection are to be treated as ‘any’. Supports IPV4, IPV6 and CIDR notation.	Required

Implementation of the the ip_connection AND the ip_addr target is required for OpenC2 producers.

OpenC2 consumers MUST implement the ip_connection OR the ip_addr target. If a device or instance of the SLPF firewall supports the ip_connection target, then the ip_addr target is OPTIONAL. Conversely, if the ip_addr target is implemented, then the ip_connection is OPTIONAL.

2.1.3 Command Arguments

Arguments provide additional precision to a command by including information such as how, when or where a command is to be executed. Table 3 summarizes the command arguments as they relate to SPLF firewall functionality and identifies which are required to implement for at least one action/target pair.

Table 3. SLPF Firewall options

	Type	Description	Req/Opt
response	string	Indicate the type of response required for the action.	Required
start-time	datetime	The specific date/time to initiate the action. Implementations SHOULD use UTC to ensure consistency across different time zones. Unspecified start time defaults to ‘now’	Required
end-time	datetime	The specific date/time to end the action. Implementations SHOULD use Universal Time	Required

		(UTC) to ensure consistency across different time zones. Unspecified end-time defaults to 'never'	
duration	int	[=] seconds.	Required

The semantics of start-time/end-time/duration are:

- If none are specified then the start time is now and the end time is never and the duration is infinity
- Only two of the 3 are allowed on any given command and the third is derived from the equation end-time = start-time + duration
- If only start time is specified then end-time is never and duration is infinity
- If only end time is specified then start-time is now and duration is derived
- If only duration is specified then start-time is now and end-time is derived

2.2 Actuator data model

This section defines the data model for the profile. The ACTUATOR field within an OpenC2 message command is identified with the actuator profile, and this section defines the actuator-specifiers and actuator-arguments needed for additional precision.

- The SPECIFIERS provide information with respect to a particular actuator to increasing levels of precision.
- ACTUATOR ARGUMENTS provide information with respect to how the ACTUATOR is to execute the action.

2.3 Actuator Specifiers

An ACTUATOR is the entity that provides the functionality and performs the action. The ACTUATOR executes the ACTION on the TARGET. In the context of this profile, the actuator is the SPLF firewall and the presence of one or more specifiers further refine which actuator(s) shall execute the action.
Table 4 identifies the specifiers that are applicable to the SPLF firewall actuator. Section 3 provides sample commands with the use of specifiers.

Table 4. SLPF Firewall Specifiers

2.3.1 Type Name: SlpfSpecifiers

Base Type: Map

Firewall Specifier	Type	Description
hostname	String	RFC 1123 hostname (can be a domain name or IP address) for a particular device with firewall functionality
Named Group	string	User defined collection of devices with firewall functionality
Asset_id	string	Unique identifier for a particular firewall

2.4 Actuator Arguments

The command arguments that are defined in this specification and apply to SLPF firewalls are presented in Table 5. These options provide detail on how the action is executed.

Table 5. Action Specific SLPF Firewall Actuator Options

2.4.1 Type Name: SpffArgs

Base Type: Map

Specifier		Description
Drop	Boolean	Stop processing and do not send a notification to the source of the packet.
Reject	Boolean	Stop processing and send a notification to the source of the packet.
Complete	Boolean	Stop processing and send a false acknowledgment to the source that the processing was completed.
Running	Boolean	Normal operations assumes any change to a device are to be implemented as persistent changes. Setting the running modifier to TRUE results in a change that is not persistent in the event of a reboot or restart.
direction	string	Possible settings are ingress, egress or both. The default value is both. Ingress applies the allow to incoming traffic only. Egress applies to outbound. Entities that do not support directionality MUST return a 501 error code when they receive ingress or egress as the option and MAY include 'Directionality not supported' in the error description.
prepend	boolean	If set to TRUE, then the new rule is placed at the beginning of the rule list. If set to FALSE, then the new rule is appended to the list. The default value is FALSE.
rule-number	integer	A command option defined in this specification. An integer. Can only be sent if prepend is NOT sent. Specifies the number of the rule within a list, typically used in a top-down rule list.

> **Editor's Note** - For now, both the 'prepend' option vs the 'rule_number' option are present. Discussion regarding whether or not they are redundant is ongoing.

2.5 OpenC2 Responses

Response messages originate from the actuator and are informative rather than a command or request that the recipient execute some task(s).
The status codes presented in Table 6 apply to all actions and identifies which responses are required to implement for at least one command.
The formats and what is expected in the response for specific commands will be captured in section 2.3 and examples will be provided in section 5, Appendix A. Responses associated with required actions MUST be implemented. Implementations that include optional actions MUST implement the Responses associated with the implemented action.

Table 6. Response Codes

Status Code	Status Text	
102	Processing. Command received but action not necessarily complete	Optional
200	OK	Required
400	Unable to process command, parsing error	Required
401	Authentication or authorization failure	Optional
403	Forbidden	MTI
500	Server Error	Optional
501	Not implemented	Required

> **Editor's Note** -No use cases have been defined for 401 and 403 at this time, but the two response codes were left in for this CSD
> **Editor's Note** -The response descriptions and the 'response complete' need to be added here after consensus is reached on the openc2 commands

2.6 OpenC2 Commands

An OpenC2 command consists of an ACTION/TARGET pair and associated specifiers and arguments. This section will enumerate the allowed commands, identify which are required and present the associated responses.
Table 7 defines the commands allowed by the firewall profile indicates which are required. The subsequent subsections provide the property tables applicable to each OpenC2 command.

Table 7. Command Matrix

	Allow	Deny	Query	Delete
	required	required		
	required	required		
			required	
			optional	optional

2.6.1 'Allow'

Table 8 summarizes the command options that apply to all of the commands consisting of the 'allow' action and a valid target type.
Upon receipt of an unsupported command-option, SLPF firewalls MUST respond with the 501 error code and SHOULD respond with "Option not supported" in the error description.
Products that send 'allow target' commands and support the rule_number option:

- MUST support the ap-slpf target type as defined in section 2.3.3.2
- SHOULD populate the command options field with "response-type="complete" "
- MUST populate the command-id option if the "response-type="complete" "
- MAY populate the command-id option regardless of the presence or lack of other options.

Products that receive 'allow target' commands and support the prepend option:

- MUST support the ap-slpf target type as defined in section 2.3.3.2
- MUST return the rule number assigned to the rule in the ap-slpf object if the "response-type= "complete" " option is populated.

Table 8. Command Arguments

Option	Req/Opt	Description/effect
response	Required	Indicates the type of response required from the firewall. Valid response types are Ack, Complete and None. The default is none.
start-time	Required	The time when the allow is to take effect. Date Time data format. Implementations SHOULD use UTC.
end-time	Required	The time when the allow rule is to be removed. Date Time data format. Implementations SHOULD use UTC.
duration	Required	The amount of time after the start-time for the allow to be in effect, specified in [=] seconds. In the absence of the start-time option, the duration starts from the time the command is processed. In the event of a conflict the end-time option (if present) takes precedence.
respond-to	Optional	Identifies where the firewall is to send its response.
running	Optional	A command option defined in this specification. Setting to TRUE results in an ephemeral allow, i.e., the traffic will be allowed while the firewall is operating, but in the event of a restart, power down, or recovery, the allow rule will not be retained. If the running option is set to FALSE, then allow commands SHOULD be persistent, i.e., the rule should remain

		in place in the event of a restart or similar event. Default value is FALSE.
direction	Optional	A command argument defined in this specification. Possible settings are ingress, egress or both. The default value is both. Ingress applies the allow to incoming traffic only. Egress applies to outbound. Entities that do not support directionality MUST return a 501 error code when they receive ingress or egress as the option and MAY include 'Directionality not supported' in the error description.
rule-number	Optional	A command option defined in this specification. A boolean. If set to FALSE (default) the allow rule is appended to the rule set. If set to TRUE, the allow rule is prepended to the rule set.

The valid target types, associated specifiers, and options are summarized in sections 2.2.1.1 through 2.2.1.2. Sample commands are presented in appendix A.

2.6.2 'Allow ip-connection'

The 'allow ip_connection' command is required for openc2 producers implementing the SLPF firewall.
If the 'allow ip_addr' target is not implemented, then SLPF consumers MUST implement the 'allow ip-connection' command. Otherwise it is OPTIONAL.
The command permits traffic that is consistent with the specified ip_connection. A valid 'allow ip-connection' command has at least one property of the ip_connection populated and may have any combination of the five properties populated. An unpopulated property within the the ip_connection target must be treated as an 'any'.
Products that do not implement the 'allow ip_connection' command MUST respond with the 501 response code and SHOULD respond with 'Target type not supported' in the error description

2.6.3 'Allow ip-addr'

The 'allow ip_addr' command is required for openc2 producers implementing the SLPF firewall.
If the 'allow ip_connection' target is not implemented, then SLPF consumers MUST implement the 'allow ip_addr' command. Otherwise the 'allow ip-addr' command is OPTIONAL.
The command permits traffic as specified by the ip_addr property and may be an IPV4 or IPV6 address. The ip-addr supports CIDR notation. The address specified in the ip_addr MUST be treated as a source OR destination address.
Products that do not implement the 'allow ip-addr' command MUST respond with the 501 response code and SHOULD respond with 'Target type not supported' in the error description.

2.6.4 'Deny'

'Deny' can be treated as mathematical complement to 'allow'. With the exception of three additional actuator-arguments, the targets, specifiers, options and corresponding responses are identical to the two 'allow' commands. Table 9 summarizes the command arguments that apply to all of the commands consisting of the 'deny' action and valid target type.

Table 9. Deny arguments

Option	Req/Opt	Description/effect
drop	Required	A command option defined in this specification. Traffic meeting the criteria of the target specifier(s) is dropped with no other processing. Default is Drop
reject	Required	A command option defined in this specification. Traffic meeting the criteria of the target specifier(s) is dropped and an ICMP host unreachable (or equivalent) is sent to the source address
complete	Optional	A command option defined in this specification. Traffic meeting the criteria of the target specifier(s) is dropped and receipt of the packet is sent to the source address, i.e. a false acknowledgement
response	Required	Indicates the type of response required from the firewall. Valid response types are Ack, Complete and None. The default is none.
start-time	Required	The time when the deny is to take effect. Date Time data format.
end-time	Required	The time when the deny rule is to be removed. Date Time data format.
duration	Optional	The amount of time after the start-time for the deny is to be in effect [=] seconds. In the absence of the start-time option, the duration starts from the time the command is processed. In the event of a conflict the end-time option (if present) takes precedence.
respond-to	Optional	Identifies where the firewall is to send its response.
running	Required	A command option defined in this specification. Setting to TRUE results in an ephemeral deny, i.e. the traffic will be denied while the firewall is operating, but in the event of a restart, power down or recovery, the rule will not be retained. Default value is FALSE
direction	Optional	A command option defined in this specification. Possible settings are ingress, egress or both. The default value is both. Ingress applies the allow to incoming traffic only. Egress applies to outbound. Entities that do not support directionality MUST return a 501 error code when they receive ingress or egress as the option and MAY include 'Directionality not supported' in the error description.

prepend	Optional	A command option defined in this specification. A boolean. If set to FALSE (default) the allow rule is appended to the rule set. If set to TRUE, the allow rule is prepended to the rule set.

Upon receipt of a command with an ARGUMENT that is not supported by the actuator, actuators MUST respond with the 501 error code and SHOULD respond with 'Option not supported' in the error description.
Products that send 'deny target' commands and support the rule-number option:

- MUST support the ap-slpf target type as defined in section 2.3.3.2
- SHOULD populate the command options field with 'response-type="complete" '
- MUST populate the command-id option if the 'response-type="complete"
- MAY populate the command-id option regardless of the presence or lack of other options.
- MAY populate the command options field with 'rule-number = INT' where INT is an integer.
- MUST populate the command options field with 'response-type="complete" ' if the 'rule-number = INT ' option is populated.

Products that receive 'deny target' commands and support the rule-number option:

- MUST support the ap-slpf target type as defined in section 2.3.3.2
- MUST return the rule number assigned in the ap-slpf object if the 'response-type= "complete" ' option is populated.
- MUST use the rule number provided if the 'rule-number = INT ' option is populated
- If the rule number is currently in the rule set, then MUST respond with the 501 error code and SHOULD respond with 'Rule number currently in use' in the error description

2.6.5 'Query'

The valid target types, associated specifiers, and options are summarized in sections 2.2.3.1 through 2.2.3.2. Sample commands are presented in Section 5 appendix A.

2.6.6 'Query openc2'

The 'query openc2' command is used to determine the capabilities of the actuator. SLPF firewalls MUST implement the 'query openc2' command. The specifiers for the openc2 target type are summarized in table 10.

Table 10. Specifiers for the openc2 target type

Specifier	Implement	Type	Values/Description
version	Required	number	1.0/ Version of the Language specification supported by this profile
profiles	Required	array	A two by n array where: [0,i] = name of the profile supported [1, i]= full path where the schema resides

			'Stateless-packet-filtering' MUST be included in the list.
profile-options	Required	array	A three by n array where: [0, i] = "stateless-packet-filtering" [1,i] = action target pair supported. If empty, the option identified applies to all action target pairs [2,i]= option supported for the identified action target pair
serialization	Required	list	MUST include 'json'. MAY include other serializations that are validated against the abstract schema presented in appendix B
product	Optional	string	The particular vendor product or image such as iptables, amazon, azure etc

Products that send the 'query openc2' command:

- MUST populate the command options field with 'response-type="complete" '.
- MUST populate the command-id.
- MAY populate the respond-to option.
- MUST NOT include other command arguments.
- MAY include one or more of the openc2 specifiers identified in table 2.2.1.1.

Products that receive the 'query openc2' command:

- That cannot parse or process the query openc2 command MUST respond with response code 400.
- Upon successful parsing and processing of the query openc2 command, products MUST respond with response code 200. The results field MUST be populated with the openc2 target type and associated specifiers identified in the command.
- If no specifiers were identified in the openc2 command, then the results field MUST contain all of the specifiers identified in Table 10. Refer to section 4 for sample commands.

2.6.7 'Query ap-slpf-query'

The 'query ap-slpf' command is used to determine the current settings of the firewall. Implementation of the 'query ap-slpf' command is OPTIONAL. The ap-slpf target is defined in this specification. Implementations that choose to include the ap-slpf-query target type MUST import it in accordance with the procedures defined in section 2.2.6 of Version 1.0 of the OpenC2 Language Specification. The ap-slpf-query data profile are:

1. The namespace identifier is: ap-slpf-query
2. The name for the data profile is: /docs.oasis-open.org/openc2/futurepath4profiles

3. A list of the object identifiers to be imported. Permitted objects are presented in table 2.2.1.2. Implementations that choose to include the ap-slpf-query MUST include the objects listed as 'required' and MAY include the objects listed as optional.

2.6.7.1 Type Name: Spff-Target

Base Type: ArrayOf(Spff-Query-Item)

2.6.7.2 Type Name: Spff-Query-Item

Base Type: Enumerated

ID	Name	Description
1	allow-rules	Return allow-rules value in the response
2	deny-rules	Return deny-rules value in the response

	Implement	Type	Values/Description
allow-rules	Required	array	A 2xn array where [0,i] is an ip-connection or ip_addr object that is explicitly allowed by the firewall and [1,i] is the corresponding rule number. The second element of the array is optional.
deny-rules	Required	array	A 2xn array where [0,i] is an A list of ip-connection or ip_addr objects that is are explicitly denied by the firewall and [1,i] is the corresponding rule number. The second element of the array is optional.

Products that send the 'query ap-slpf' command:

- MUST populate the command options field with 'response-type="complete".
- MUST populate the command-id.
- MAY populate the respond-to argument.
- MUST NOT include other command arguments.
- MAY include one or more of the ap-slpf-query specifiers identified in table 2.2.1.2.

Products that receive the 'query ap-slpf' command:

- MUST respond with response code 400 if the command cannot be parsed or processed.
- MUST respond with error code 501 and MAY respond with error description 'data model not supported' if the product does not support the ap-slpf target type
- Upon successful parsing and processing of the 'query ap-slpf' command, products MUST respond with response code 200 and populate the results field with the ap-slpf-query target type and associated specifiers identified in the command.

- If one or more ip-connection objects were explicitly listed in the allow-rules specifier, then the actuator returns the ip-connection if it is explicitly allowed. An empty string implies that the ip-connection is not allowed.

- If one or more ip-connection objects were explicitly listed in the deny-rules specifier, then the actuator returns the ip-connection if it is explicitly denied. An empty string implies that the ip-connection is not denied.
- If no ip-connection objects were explicitly listed in the allow-rules specifier, then the actuator MUST return the complete list of ip-connection objects that are explicitly allowed.
- If no ip-connection objects were explicitly listed in the deny-rules specifier, then the actuator MUST return the complete list of ip-connection objects that are explicitly denied.
    - If no specifiers were identified in the query ap-slpf-query command, then the results field MUST return the entire allow and deny list.

Refer to section 4 for sample commands.

2.6.8 'Delete'

The ap-slpf is the only valid target type for the delete action. The associated specifiers, and options are summarized in section 2.2.4.1. Sample commands are presented in section 4.

2.6.9 'delete ap-slpf'

The 'delete ap-slpf' command is used to remove a firewall rule rather than issue another allow or deny to counteract the effect of an existing rule. Implementation of the 'delete ap-slpf' command is OPTIONAL. Products that choose to implement the 'delete ap-slpf' command MUST implement the ap-slpf target type described in section 2.3.3.1.
Products that send the 'delete ap-slpf' command MAY populate the command options field with 'response-type="complete".
Products that send the 'delete ap-slpf' command MAY populate the command-id.
Products that send the 'delete ap-slpf' command MAY populate the respond-to argument.
The 'delete ap-slpf' command MUST NOT include other command arguments.
Products that send the 'delete ap-slpf' command MUST include exactly one deny-rule OR allow-rule as described in table 2.2.1.2.
The deny-rule OR allow-rule MUST include the rule number and MAY include the corresponding ip_connection or ip_addr.
Products that receive, but cannot parse or process the 'delete ap-slpf' command MUST respond with response code 400.
Products that receive but do not support the ap-slpf target type MUST respond with error code 501 and SHOULD respond with error description 'target type not supported'
Upon successful parsing of the 'delete ap-slpf' command and subsequent removal of the corresponding rule, products MUST respond with response code 200.
Upon successful parsing of the 'delete ap-slpf' command and failure to remove the corresponding rule, products MUST respond with response code 500 and SHOULD respond with error description 'firewall rule not removed or updated'.
Refer to section 4 for sample commands.
This page intentionally left blank.

3 Conformance statements

> Editor's Note - This section is a placeholder. The conformance clauses stated within the body of the profile will be repeated here once general consensus is reached.<Note: need to define "support" - behavior that must occur as a result of executing commands>

Conformant implementations of OpenC2 Firewall Functions:

- MUST support OpenC2 commands, responses, and alerts as defined in Section 4.
- MUST implement JSON serialization of the commands, responses and alerts that are consistent with the syntax defined in the OpenC2 Language Specification.
- MAY implement any serialization that provides a one to one mapping of the data elements as defined in the abstract schema presented in the Language Specification to a format that is consistent with the syntax defined in Appendix B OpenC2 SCHEMA – FIREWALL FUNCTIONS.
- MUST implement the actuator data model as presented in the abstract schema presented in Appendix B.

---

4 References

> **Editor's Note** - This section is a placeholder. The normative and non-normative references will be populated once general consensus is reached.

1. https://www.cisco.com/c/en/us/products/security/firewalls/what-is-a-firewall.html
2. https://www.juniper.net/uk/en/solutions/software-defined-secure-networks/next-gen-firewall
3. https://www.paloaltonetworks.com/cyberpedia/what-is-a-firewall

This page intentionally left blank.

---

5 Appendix A. Sample commands

> **Editor's Note** - This section is a placeholder. he syntax of the sample commands all must be reworked in accordance with the changes that occured in the language specification

This section will summarize and provide examples of OpenC2 commands as they pertain to firewalls. The sample commands will be encoded in verbose JSON, however other encodings are possible provided the command is validated against the schema presented in Appendix A. Examples of corresponding responses and/or alerts will be provided where appropriate.
The samples provided in this section are for illustrative purposes only and are not to be interpreted as operational examples for actual systems. Within the scope of this document, a #

character indicates a comment, however it should be noted that OpenC2 itself does not support comments within a command.

5.1 Deny and Allow

Deny and allow are mandatory to implement and can be treated as mathematical complements of each other. Unless otherwise stated, the example targets, specifiers, modifiers and corresponding responses are applicable to both actions.

```
```
Block a particular connection within the domain and do not send a host
unreachable
{
    "action": "deny",
    "target": {
        "ip_connection": {
            "layer4_protocol": "TCP",
            "src_addr": {"1.2.3.4"},
            "src_port": {10996},
            "dst_addr": {198.2.3.4},
            "dst_port": {80}
        }
    },
    "actuator": {
        "firewall": {
            "asset_id": "30"
                Options:{drop}
        }
    },
    "command-options": {
        ,
        "start_time": "2016-11-25T08:10:31-04:00",
        "duration": 600,
        "command_id": "fw17_8675309"
    }
}
```
```
```
# Block all ftp data transfers from hosts, send false acknowledgement and
request ack. Note that the five-tuple is incomplete
{
    "action": "deny",
    "target": {
        "type": ":ip_connection",
        "specifiers": {
            "Layer4Protocol": "TCP",
            "src-port": 21
        }
    }
    "actuator": {
```

```
        "type": "openc2:firewall",
        "specifiers": {endpoint},
        "options":{complete}
    },
    "command-options": {
        {"id":"UUID=123e4567-e89b-12d3-a456-426655440000"}
        {response=”Ack”}
    }
}
# Note that the response was requested and all endpoints that can execute the
command should.
```
```
# In this case, one of the endpoints successfully issued the deny but the
endpoint located at 198.51.100.17 failed
{
    response
        {Source: ip-addr=198.51.100.17}
        {cmdref=123e4567-e89b-12d3-a456-426655440000 }
        {statuscode=200}
}
{
    response
        {Source: ip-addr=198.51.100.18}
        {cmdref=123e4567-e89b-12d3-a456-426655440000 }
        {statuscode=400}
}
```
```
# Allow ftp data transfers to a particular ip address from any host. Note
that the five-tuple is incomplete
{
    "action": "allow",
    "target": {
        "type": "ip-connection",
        "specifiers": {
            "Layer4Protocol": "TCP",
            "dst-addr": 198.51.100.17
            "src-port": 21
        }
    }
      "command-options": {
            "id":"UUID=123e4567-e89b-12d3-a456-426655440000"
            “response”:”Ack”
        }
    }
}
#
{
    response
        {Source: openc2:ip-addr=1.2.3.4}
```

```
        {cmdref=123e4567-e89b-12d3-a456-426655440000 }
        {statuscode=200}
}
```

5.2 Update

Implementation of the Update action is optional. Update is intended for the device to process new configuration files, software updates, patches, policy updates etc. The update action is a compound action in that all of the steps required for a successful update (such as download the new file, install the file, reboot etc.) are implied. File is the only valid target type for Update.

```
# instructs the firewalls to acquire a new configuration file. Note that all
network based firewalls will install the new update because no particular
firewall was identified. Host based firewalls will not act on this because
network firewalls were identified as the actuator.
{
    "action": "update",
    "target": {
        "file": {
            "parent_directory": {
            "path":"\\\\someshared-drive\\somedirectory\\configurations"},
            "name": "firewallconfiguration.txt"
        }
    },
    "actuator": {
        "openc2:firewall": {
            "named-group":"network"
        }
    }
    “Command-options”{
        Command-id:123e4567-e89b-12d3-a456-426655440000
}
}
```
```
Note the responses.  One of the devices successfully updated the
configuration file.  Another device responded with an error because it does
not support the ‘update file’ command
{
    response
        {Source: openc2:ip-addr=1.2.3.4}
        {cmdref=123e4567-e89b-12d3-a456-426655440000 }
        {statuscode=200}
}
{
    response
        {Source: openc2:ip-addr=1.2.3.5}
        {cmdref=123e4567-e89b-12d3-a456-426655440000 }
        {statuscode=501}
```



```
          {statustext:”command not supported”}
}
```
```
# Instructs any firewall running a particular software load to install a
software upgrade
{
    "action": "update",
    "target": {
        "file": {
            "parent_directory": {
                "path": "\\\\someshared-drive\\somedirectory\\so"
            },
            "name": " version2.offirewallsoftware.exe"
        }
    },
    "actuator": {
        "openc2:firewall": {
            "x-tagID": "firewallcompanyversion2.1"
        }
    }
}
```

7.4 Query

The query action is used to gather some set of information from the firewall. There are two valid target types for the query command; ‘openc2’ and ‘ap-slpf-query’.
Implementation of query openc2 is required. The query openc2 command is intended to enable the openc2 producer to determine the capabilities of the actuator. The query openc2 command can also be used to check the status of the actuator.

```
# This thread illustrates the use of query openc2 to verify that the actuator
# is functioning and/or determine the version of the language specification.
# Note that the specifier is identified in the command but the value is not
# populated.
{
    “Id”=12345
    "action": "query"
    "target": {
            "openc2" {
                    Version:
        }
    }
“Options”{
Response_type=complete}
}

#  The corresponding response would be:
#  Case one, things are OK
    “Id”:12345
    “Status”:200
```

```
    Results{
                    Type: openc2{
                    “Version”: 1.0
}
}
#  Case two, error state
    “Id”:12345
    “Status”:400
    Results{}


# This thread illustrates the use of the query openc2 command to determine
the
# the profiles supported and version of openc2.
{
    “Id”=12345
    "action": "query"
    "target": {
            "openc2" {
                    Version:
                    Profiles:
            }
    }
“options”{
Response_type=complete}
}

# The corresponding response would be:
# Case one, things are OK
    “Id”:12345
    “Status”:200
    Results{
                    Type: openc2{
                    “Version”: 1.0
                    “profiles”: stateless-packet-filtering, http://some.url.org
}
}
#  Case two, error
    “Id”:12345
    “Status”:400
    Results{}

# Case two:  We know the version and profile, but want to know the options
#  profile-options is present but not populated
{
    “Id”=12345
    "action": "query"
    "target": {
            "openc2" {
                    Profile-options
```

```
            }
    }
“options”{
Response_type=complete}
}

The corresponding response would be:
                    # response one, things are OK
    “Id”:12345
    “Status”:200
    Results{
                    Type: openc2{
Profile-options:
                    stateless-packet-filtering, deny domain-name, complete
                    stateless-packet-filtering, deny domain-name, respond-to
                    stateless-packet-filtering, deny hostname, complete
                    stateless-packet-filtering, deny hostname, respond-to
}
}
#  Response two, error
    “Id”:12345
    “Status”:400
    Results{}

# Case three:  In this case, we don’t know anything and want a complete
report from the actuator.
#  None of the specifiers for the openc2 target type are present, so the
actuator responds with all of the specifiers populated
{
    “Id”=12345
    "action": "query"
    "target": {
            "openc2" {
            }
    }
“options”{
Response_type=2}
}

#The corresponding response would be:
# response one, things are OK
    “Id”:12345
    “Status”:200
    Results{
        Type: openc2{
            Version: 1.0
              “Profiles”:
stateless-packet-filtering, http://some.url.org
our-next-oasis-profile, http://someother.url.org
   Profile-options:
```

```
                              stateless-packet-filtering, deny domain-name,
complete
                              stateless-packet-filtering, deny domain-name,
respond-to
                              stateless-packet-filtering, deny hostname,
complete
                              stateless-packet-filtering, deny hostname,
respond-to
                              our-next-oasis-profile, scan directory, act-
specific-option
                              our-next-oasis-profile, scan file, actuator-
specific-option
# note in the next entry, we had an optional action target pair, but no
options associated with it
                    our-next-oasis-profile, contain username,
# note in the next entry, we have a command option for all profiles for all
actions
                    ,,respond-to
Serialization:
Json
Cbor
XML

}
}
                        #  Response two, error
    “Id”:12345
    “Status”:400
    Results{}
Implementation of query ap-slpf-query command is optional.  The query ap-slpf-
query is intended to enable the openc2 producer to determine the current
state of the actuator.
# Verify that the firewalls are blocking all outbound ftp traffic and TELNET
traffic
{
    “Id”=12345
    "action": "query"
    "target": {
            "ap-slpf-query" {
                    Deny-rules:
                        {specifiers
{src-port:21}
}
                        {specifiers
                                {dst-port:23}
                    }
        }
    }
“Options”{
Response_type=complete}
```

```
}

#  The corresponding response would be:
#  Case one, things are OK and both ip-connections are denied
    “Id”:12345
    “Status”:200
    Results{
                    Type: ap-slpf-query{
                            {src-port:21}
                        {dst-port:23}
}
}
#  Case two, things are OK.  Telnet is being denied but outbound ftp is
notboth ip-connections are denied
    “Id”:12345
    “Status”:200
    Results{
                    Type: ap-slpf-query{
                            {dst-port:23}
}
}
#  Case three, not supported
    “Id”:12345
    “Status”:501
    Results{}
```

---

6 Appendix B. OpenC2 SCHEMA – FIREWALL FUNCTIONS

> **Editor's Note** - This section is a placeholder.