CIP Inspector

CIP Inspector Overview

Type	Inspector (service)
Usage	Inspect
Instance Type	Multiton
Other Inspectors Required	`stream_tcp`
Enabled	`false`

The Common Industrial Protocol (CIP) is an application protocol that supports industrial automation applications. EtherNet/IP (ENIP) is an implementation of CIP that is used on Ethernet-based networks.

The `cip` inspector detects CIP and ENIP traffic running on TCP or UDP and sends it to the intrusion rules engine. You can use CIP and ENIP keywords in custom intrusion rules to detect attacks in CIP and ENIP traffic.

Note In Snort 3, the `cip` inspector does not support CIP application detectors. To implement CIP application detection, you can create and import custom CIP intrusion rules and enable the appropriate IPS rules. For more information, see the Snort 3 configuration documentation for your management application.

Best Practices for Configuring the CIP Inspector

Consider the following best practices when configuring the `cip` inspector:

- You must add the default CIP detection port 44818 and any other CIP ports in the `binder` inspector.
- We recommend that you use an intrusion prevention action as the default action for your access control policy.
- To detect CIP and ENIP applications, you must enable the `cip` inspector in the corresponding custom network analysis policy.
- To block CIP or ENIP application traffic using access control rules, ensure that the normalizer inspector and its inline mode option are enabled (the default setting) in the corresponding network analysis policy.
- To drop traffic that triggers `cip` inspector rules and CIP intrusion rules, ensure that **Drop when Inline** is enabled in the corresponding intrusion policy.
- The `cip` inspector does not support an access control policy default action of either of the following:
  - **Access Control: Trust All Traffic**
  - **Access Control: Block All Traffic**
- The `cip` inspector does not support application visibility for CIP applications, including network discovery.

CIP Inspector Parameters

CIP TCP port configuration

The `binder` inspector defines the CIP TCP port configuration. For more information, see the Binder Inspector Overview.

Example:

```
[
    {
        "when": {
                    "role": "server",
                    "proto": "tcp",
                    "ports": "44818"
        },
        "use": {
                    "type": "cip"
        }
    }
]
```

embedded_cip_path

Determines whether the inspector checks the embedded CIP connection path.

Type: string

Valid values:

- `"false"`
- CIP path enclosed in double quotation marks, for example, `"0x2 0x36"`.

Default value: `"false"`

unconnected_timeout

Sets the default unconnected timeout in seconds. When a CIP request message does not contain a protocol-specific timeout value and the maximum number of concurrent unconnected requests per TCP connection is reached, the system times the message for the number of seconds specified by this parameter. When the timer expires, the message is removed to make room for future requests.

When you specify 0, all traffic that does not have a protocol-specific timeout configured times out first.

Type: integer

Valid range: `0` to `360`

Default value: `300`

max_unconnected_messages

Sets the maximum number of concurrent unconnected CIP messages per TCP connection. If the system reaches the maximum number of concurrent requests that can go unanswered, the system closes the connection.

Type: integer

Valid range: `1` to `10000`

Default value: `100`

max_cip_connections

Sets the maximum number of simultaneous CIP connections allowed by the system per TCP connection.

Type: integer

Valid range: `1` to `10000`

Default value: `100`

CIP Inspector Rules

Enable the `cip` inspector rules to generate events and, in an inline deployment, drop offending packets.

Table 1: CIP Inspector Rules

GID:SID	Rule Message
148:1	CIP data is malformed
148:2	CIP data is non-conforming to ODVA standard
148:3	CIP connection limit exceeded. Least recently used connection removed
148:4	CIP unconnected request limit exceeded. Oldest request removed

CIP Inspector Intrusion Rule Options

cip_attribute

Detection parameter to match the CIP attribute.

Type: interval

Syntax: `cip_attribute: <range_operator><positive integer>;` or `cip_attribute: <positive integer><range_operator><positive integer>;`

Valid values: A set of one or more integers between 0 and 65535, and a `range_operator` as specified in the Table 2: Range Formats.

Examples: `cip_attribute: <100;`

cip_class

Detection parameter to match the CIP class.

Type: interval

Syntax: `cip_class: <range_operator><positive integer>;` or `cip_class: <positive integer><range_operator><positive integer>;`

Valid values: A set of one or more integers between 0 and 65535, and a `range_operator` as specified in the Table 2: Range Formats.

Examples: `cip_class: <25;`

cip_conn_path_class

Detection parameter to match the CIP connection path class.

Type: interval

Syntax: `cip_conn_path_class: <range_operator><positive integer>;` or `cip_conn_path_class: <positive integer><range_operator><positive integer>;`

Valid values: A set of one or more integers between 0 and 65535, and a `range_operator` as specified in the Table 2: Range Formats.

Examples: `cip_conn_path_class: <85;`

cip_instance

Detection parameter to match the CIP instance.

Type: interval

Syntax: `cip_instance: <range_operator><positive integer>;` or `cip_instance: <positive integer><range_operator><positive integer>;`

Valid values: A set of one or more integers between 0 and 65535, and a `range_operator` as specified in the Table 2: Range Formats.

Examples: `cip_instance: <15;`

cip_req

Detection parameter to match the CIP request.

Syntax: `cip_req;`

Examples: `cip_req;`

cip_rsp

Detection parameter to match the CIP response.

Syntax: `cip_rsp;`

Examples: `cip_rsp;`

cip_service

Detection parameter to match the CIP service.

Type: interval

Syntax: `cip_service: <range_operator><positive integer>;` or `cip_service: <positive integer><range_operator><positive integer>;`

Valid values: A set of one or more integers between 0 and 127, and a `range_operator` as specified in the Table 2: Range Formats.

Examples: `cip_service: <50;`

cip_status

Detection parameter to match the CIP response status.

Type: interval

Syntax: `cip_status: <range_operator><positive integer>;` or `cip_status: <positive integer><range_operator><positive integer>;`

Valid values: A set of one or more integers between 0 and 255, and a `range_operator` as specified in the Table 2: Range Formats.

Examples: `cip_status: <250;`

Table 2: Range Formats

Range Format	Operator	Description
operator i		
	<	Less than
	>	Greater than
	=	Equal
	≠	Not equal
	≤	Less than or equal

Range Format	Operator	Description
	≥	Greater than or equal
j operator k		
	<>	Greater than j and less than k
	<=>	Greater than or equal to j and less than or equal to k