Introduction

The Cisco Event Streamer (also known as eStreamer) allows you to stream Firepower System events to external client applications. You can stream host, discovery, correlation, compliance allow list, intrusion, user activity, file, malware, and connection data from a Management Center and you can stream intrusion data from 7000 and 8000 series devices.

Note that eStreamer is not supported on NGIPSv, Firepower Services, Firepower Threat Defense Virtual, and Firepower Threat Defense. To stream events from these devices, you can configure eStreamer on the Management Center that the device reports to.

eStreamer uses a custom application layer protocol to communicate with connected client applications. As the purpose of eStreamer is simply to return data that the client requests, the majority of this guide describes the eStreamer formats for the requested data.

There are three major steps to creating and integrating an eStreamer client with a Firepower System:

1. Write a client application that exchanges messages with the Management Center or managed device using the eStreamer application protocol. The eStreamer SDK includes a reference client application.
2. Configure a Management Center or device to send the required type of events to your client application.
3. Connect your client application to the Management Center or device and begin exchanging data.

This guide provides the information you need to successfully create and run an eStreamer Version 6.3 client application.

Major Changes in eStreamer Version 6.3

Added failed user login, VPN user login, and VPN user logoff events in Discovery and Connection Event Record Types

Added a Best Practices section.

Using this Guide

At the highest level, the eStreamer service is a mechanism for streaming data from the Firepower System to a requesting client. The service can stream the following categories of data:

- Intrusion event data and event extra data
- Correlation (compliance) event data

- Discovery event data
- User event data
- Metadata for events
- Host information
- Malware event data

Descriptions of the data structures returned by eStreamer make up the majority of this book. The chapters in the book are:

- Understanding the eStreamer Application Protocol, page 2-1, which provides an overview of eStreamer communications, details some of the requirements for writing eStreamer client applications, and describes the four types of messages used to send commands to and receive data from the eStreamer service.
- Understanding Intrusion and Correlation Data Structures, page 3-1, which documents the data formats used to return event data generated by the intrusion detection and correlation components and the data formats used to represent the intrusion and correlation events.
- Understanding Discovery & Connection Data Structures, page 4-1, which documents the data formats used to return discovery, user, and connection event data.
- Understanding Host Data Structures, page 5-1, which documents the data formats that eStreamer uses to return full host information data when it receives a host information request message.
- Configuring eStreamer, page 6-1, which documents how to configure the eStreamer on a Management Center or managed device. The chapter also documents the eStreamer command-line switches and provides instructions for manually starting and stopping the eStreamer service and for configuring the Management Center or managed device to start eStreamer automatically.
- Data Structure Examples, page A-1, which provides examples of eStreamer message packets in binary format.
- Understanding Legacy Data Structures, page B-1, which documents the structure of legacy data structures that are no longer in use by the currently shipping product but may be used by older clients.

Prerequisites

To understand the information in this guide, you should be familiar with the features and nomenclature of the Firepower System and the function of its components in general, and with the different types of event data these components generate in particular. Definitions of unfamiliar or product-specific terms can frequently be obtained from the *Firepower eStreamer Integration Guide*.

Product Versions for Firepower System Releases

Version numbers are used throughout this guide to describe the data format for events generated by the Management Center and managed devices. The Firepower System Product Versions table lists versions for each product by major release.

Table 1-1 Firepower System Product Versions

Release	Management Center Version	Managed Device Version
3D System 5.0	Management Center 5.0	5.0
3D System 5.1	Management Center 5.1	5.1
3D System 5.1.1	Management Center 5.1.1	5.1.1
3D System 5.2	Management Center 5.2	5.2
3D System 5.3	Management Center 5.3	5.3
Firepower System 5.3.1	Management Center 5.3.1	5.3.1
Firepower System 5.4	Management Center 5.4	5.4
Firepower System 6.0	Management Center 6.0	6.0
Firepower System 6.1	Management Center 6.1	6.1
Firepower System 6.2	Management Center 6.2	6.2
Firepower System 6.2.1	Management Center 6.2.1	6.2.1
Firepower System 6.2.2	Management Center 6.2.2	6.2.2
Firepower System 6.2.2	Management Center 6.2.3	6.2.3
Firepower System 6.3.0	Management Center 6.3.0	6.3.0

Document Conventions

The eStreamer Message Data Type Conventions table lists the names used in this book to describe the various data field formats employed in eStreamer messages. Numeric constants used by the eStreamer service are typically unsigned integer values. Bit fields use low-order bits unless otherwise noted. For example, in a one-byte field containing five bits of flag data, the low-order five bits will contain the data.

Table 1-2 eStreamer Message Data Type Conventions

Data Type	Description
nn-bit field	Bit field of nn bits
byte	8-bit byte containing data of arbitrary format
int8	Signed 8-bit byte
uint8	Unsigned 8-bit byte
int16	Signed 16-bit integer
uint16	Unsigned 16-bit integer
int32	Signed 32-bit integer
uint32	Unsigned 32-bit integer
uint64	Unsigned 64-bit integer
string	Variable length field containing character data
[n]	Array subscript following any of the above data types to indicate n instances of the indicated data type, for example, uint8[4]

Table 1-2 eStreamer Message Data Type Conventions (continued)

Data Type	Description
variable	Collection of various data types
BLOB	Binary object of unspecified type, typically raw data as captured from a packet

IP Addresses

The Cisco database stores IPv4 and IPv6 addresses in the same fields in a BINARY format. To get IPv6 addresses, convert to hex notation, for example: `20010db80000000000000000000004321`. The database follows the RFC for storing IPv4 addresses by filling in bits 80-95 with 1's, which yields an invalid IPv6 address. For example, the IPv4 address 10.5.15.1 would be stored as `00000000000000000000FFFF0A050F01`.

Best Practices

When working with eStreamer, Cisco recommends the following for best use of the API.

Design

- Consider using the Cisco pluggable eStreamer client written in Python as a foundation to your client so that all you need to build is a plug-in to format data for your SIEM's schema.
- Build your eStreamer client to support everything the API can provide, as every bit of the schema is important to at least some small part of the customer base.
  - Understand the message structure – get to know the eStreamer Integration Guide.
  - Spend time getting the records defined in metadata and code structures – a huge part of this is being able to parse the messages.
  - Understand how metadata works in a general sense, e.g. that metadata records are sent in advance.
  - Understand the object model – how the records inter-relate and what metadata relates to what records.
- Implement robust error handling and logging so that when something goes wrong, you can see the message and situation that caused the problem without necessarily needing to reproduce the error.
- Pick your language carefully. Parsing might not seem that computationally expensive but when there are thousands of events per second, everything counts. Compiled languages such as C, C++, Go will be faster than Python / JavaScript. The downside of such an approach is lack of portability.
- If you implement multi-threading or processing, understand that whatever handles the metadata has to process the messages in order – which must include out-of-order delivery corrections.
- Look at the existing eStreamer implementations to see how others have accomplished your goals in the past. Some resources:
  - https://splunkbase.splunk.com and search for eStreamer
  - https://software.cisco.com/download/home/ next to "Select a Product" select "Browse All", then "Security", followed by "Firewalls", "Firewall Management", "Firepower Management Center Virtual Appliance", then "Firepower System Tools and APIs".
  - https:/community.cisco.com and search for "eNcoreCLI".

- Make sure to work with the Cisco Security Technical Alliance team to keep up with changes to eStreamer and other aspects of integrating with Cisco Firepower. You can contact them at ask-csta-pm@cisco.com

Testing

- When Cisco introduces a new version of Firepower, promptly test your client against it to make sure the data collected by your client does not change.
- Have a good test bed so you can test easily and frequently.
- If you prefer to not build your own test bed, use the dcloud sandbox test bed. The Cisco Security Technical Alliance will provide resources to assist in setting up and using it. Dcloud is free and enables comprehensive testing. However, it is not necessarily complete for your use and does not have 100% event coverage. Also, instances are only available for short periods of time. For more information about dcloud, go to https://dcloud2-rtp.cisco.com