

IBM Common Data Provider for z Systems: Elastic Stack (Elasticsearch®, Logstash®, Kibana®) Integration Example



This document can be found on the web at www.ibm.com/support/techdocs

Version Date: August 2017

Mike Bonett

IBM - Washington Systems Center (bonett@us.ibm.com)

Special Notices

This document reflects the IBM Washington Systems Center's understanding on integrating the IBM Common Data Provider for z Systems (CDPz) with the Elastic Stack (Elasticsearch, Logstash, Kibana). It was produced and reviewed by the members of the IBM Washington Systems Center organization. This document is presented "As-Is" and IBM does not assume responsibility for the statements expressed herein. It reflects the opinions of the IBM Washington Systems Center. These opinions are based on hands on experiences with the IBM Common Data Provider for z Systems. If you have questions about the contents of this document, please direct them to Mike Bonett (bonett@us.ibm.com).

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Trademarks

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States and/or other countries: CICS, IBM, NetView, System z, WebSphere, z/OS, z/VM, z Systems. A full list of U.S. trademarks owned by IBM may be found at <http://www.ibm.com/legal/copytrade.shtml>.

Elasticsearch is a trademark of Elasticsearch BV, registered in the U.S. and in other countries

Kibana is a trademark of Elasticsearch BV, registered in the U.S. and in other countries

Logstash is a trademark of Elasticsearch BV, registered in the U.S. and in other countries

Splunk is a trademark of Splunk, Inc., registered in the U.S. and in other countries.

Microsoft, Windows, and the Windows logo are registered trademarks of Microsoft Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

LINUX and Linux are registered trademarks of Linus Torvalds.

Java and all Java-based trademarks and logos are registered trademarks of Oracle and/or its affiliates.

Other company, product and service names may be trademarks or service marks of others.

Acknowledgements

Many thanks to the following people for reviewing this paper and/or providing input to the process:

- Mike Sine, IBM Washington Systems Center

Contents

Introduction	4
Elastic Stack integration components.....	4
An example integration environment.....	6
Installing and configuring Logstash.....	6
Installing and-configuring the ELK ingestion kit.....	6
Configuring the IBM Common Data Provider for z Systems.....	8
Results.....	11
Summary	13

Introduction

The IBM Common Data Provider for z Systems (which may be referred to as “CDPz” in the document) provides a streamlined method of capturing z/OS performance and operational data for use by analytics products. The data sources CDPz can capture and forward to the target include:

- SYSLOG/OPERLOG
- JOBLOGs, with custom mapping of WebSphere and CICS JOBLOGs
- SMF Records
- NetView for z/OS logs
- Unix System Services syslogd and files

CDPz can send the data to any target that can receive data over an IP port. It provides custom integration for the following software products:

- IBM Operations Analytics for z Systems
- The Elastic Stack (Elasticsearch, Logstash, Kibana)
- Splunk

Details on CDPz can be found at

https://www.ibm.com/support/knowledgecenter/SSGE3R_1.1.0/com.ibm.cdpz.doc/welcome.html

This paper will walk through an example of integrating the Common Data Provider for z Systems with the Elastic Stack, using the z/OS SYSLOG as the data source to be sent.

Elastic Stack integration components

The Elastic Stack (formerly known as the ELK stack) is a set of open source products that integrate to provide capture, search, analysis and visualization of data. The products are:

- Logstash – used to capture, parse, format, and forward data to other products for ingestion.
- Elasticsearch – a full-text search and analytics engine that can receive data from various sources, including Logstash indexes the data to support searching and analytics against the data.
- Kibana – and graphical user interface for building views and dashboards of the data provided by Elasticsearch.

Details on the Elastic Stack can be found at <https://www.elastic.co/products>

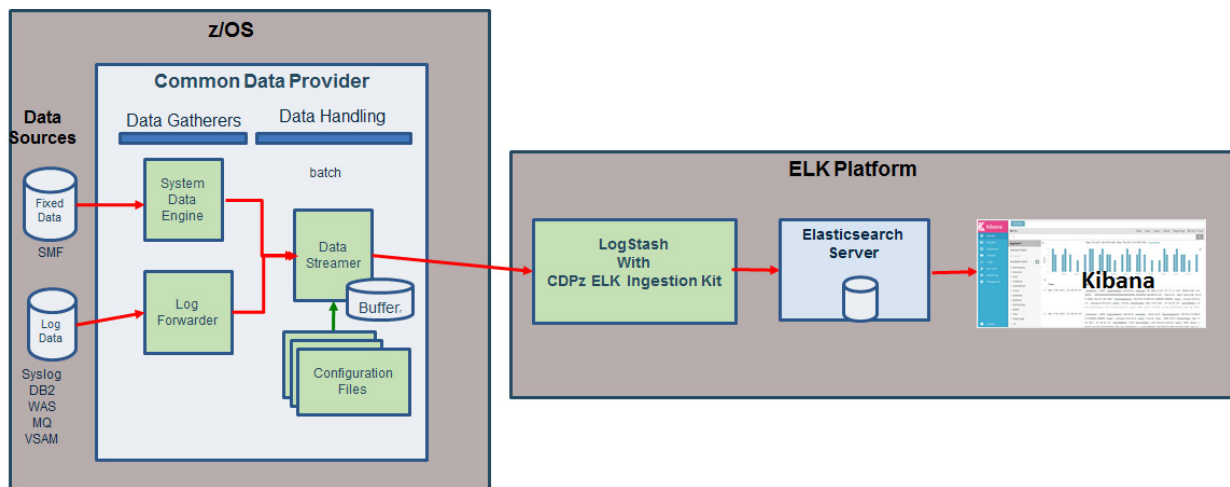
CDPz, output is integrated with the Elastic Stack in the following manner:

- CDPz forwards the desired data to a Logstash instance.
- CDPz provides an ELK Ingestion Kit. This is a set of Logstash input, filtering, mapping, and output files to prepare and forward the data to an Elasticsearch server. The Ingestion kit is installed on the same platform as the Logstash instance receiving the CDPz data.

On the z/OS side, the System Data Engine, Log Forwarder and Data Streamer components of CDPz must be configured appropriately and started (they run as started tasks or batch jobs) to send data to the Data Receiver:

- If SMF or customer data is to be forwarded, the System Data Engine must be used to access those data sources.
- If SYSLOG, OPERLOG, or data from NetView, CICS, or JOBLOGs is to be forwarded, the Log Forwarder must be used to access those data sources.
- The Data Streamer will receive the data from the System Data Engine.

An example of the architecture is shown in this picture:



The steps to enable this architecture are:

- Install and configure the target Logstash instance.
- Install the CDPz ELK Ingestion Kit files onto the same platform as the Logstash instance and customize Logstash to use the files.
- Configure the Common Data Provider for z Systems components to extract, format, and forward data to Logstash.

The rest of this paper will illustrate these steps by applying them to an example implementation.

An example integration environment

To better illustrate the integration, the rest of this paper walks through an example implementation. For this example, SYSLOG data was captured and sent to a Logstash instance server running on a Linux systems platform. The Logstash instances forwards the data to a separate Linux server where Elasticsearch and Kibana are running.

Installing and configuring Logstash

Logstash is relatively easy to install, from binary distributions or package repositories. The Elastic Company provides the Logstash code installation steps at <https://www.elastic.co/guide/en/logstash/current/installing-logstash.html>

After the code is installed the main configuration steps are defining the input sources, processing to be applied against the received data, and the output sources to send the processed data. This is defined in a set of Logstash configuration files that define one or more of the following sections, in order:

- input – the source of the data (e.g. a file or a TCP/IP port)
- filter – processing to be done against the data
- output – the destination for the data that has passed through the filter stage.

For CDPz, this information will be provided in the next step.

Installing and-configuring the ELK ingestion kit

The CDPz ELK Ingestion Kit consists of a set of Logstash configuration files to process any data that CDPz might send:

- A Logstash input definition file for receiving data from CDPz
- A pair of Logstash files for each log and SMF record type supported by CDPz, to map and filter the input for Elasticsearch usage. Only the pairs for the actual data to be sent from CDPz should be used.
- A Logstash output definition file to send the data to an Elasticsearch index server, for use in the Kibana UI

The ingestion kit is packaged as a tar.gz file and must be download from the CDPz installation directory on z/OS to the server running the target Logstash instance.

Once downloaded, the file can be unzipped, and the appropriate files copied to the defined directory from which Logstash reads its configuration files. The file names have the following naming convention:

File name	
B_	Input stage
E_	Prep stage
H_	Field name annotation
N_	Timestamp resolution
Q_	Output stages

The naming allows the stages to be processed in the proper sequences, as Logstash reads the configuration files in alphabetical order. For example, to process SYSLOG data, these files are used:

- B_CDPz_Input.1sh – this defines the input port Logstash will be listening on and the format of the incoming data, which must be json. For example:

```
input {  
  tcp {  
    port => 8081  
    codec => "json"  
  }  
}
```

- E_CDPz_Index.sh – provides a filter section that adds an index field to the data (required by Elasticsearch), that will be used as part of the overall record index (which is set in the output section)
- H_SYSLOG_OR_OPERLOG.1sh – provides a filter section that maps the csv data within the last json field to column names
- N_SYSLOG_OR_OPERLOG.1sh – provides a filter section that adds a timestamp field for when the record was received by Logstash.
- Q_CDPz_Elastic.1sh – provides an output section that sends the processed data to Elasticsearch. For example:

```

output {
    elasticsearch {
        hosts => [ "10.1.1.132:9200" ]
        user => "elastic"
        password => "changemeorelse"
        action => "index"
        index => "cdp-%{[@metadata][indexname]}-%{+yyyyMMdd}"
    }
}

```

The B_CDPz_Input.1sh, E_CDPz_Index.1sh, and Q_CDPz_Elastic.1sh files would always be used; the E_, H_, and N_ files would vary based on the types of data being sent. For example, if SMF type 30 usage records were also being sent from CDPz, the H_ and N_ files for those records would also be placed into the Logstash configuration file directory.

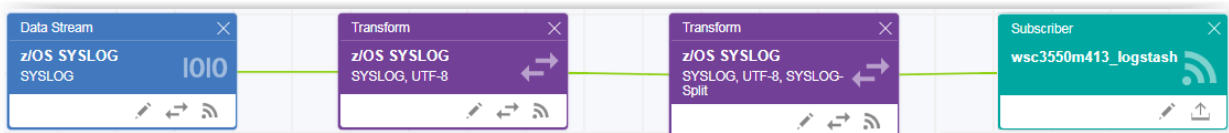
Configuring the IBM Common Data Provider for z Systems

A flow must be defined in IBM Common Data Provider for z Systems to identify:

- The data source that will be used
- Any required transformations
- The output to where the data is sent.

These actions are performed in the CDPz configuration tool, which is installed as a plugin to the z/OS Systems Management Facility (z/OSMF). The tool allows a graphical representation of the flow to be built, which then generates the files that the CDPz data streamer will read at its startup and use to collect, transform, and send data.

A graphical representation of the flow would be as follows:



This shows a Data Stream Source (SYSLOG) two transforms (UTF8 and Split) and a subscriber (wsc3550m413_logstash). Each component is defined, in order, as follows:

- Data Stream and UTF-8 transform

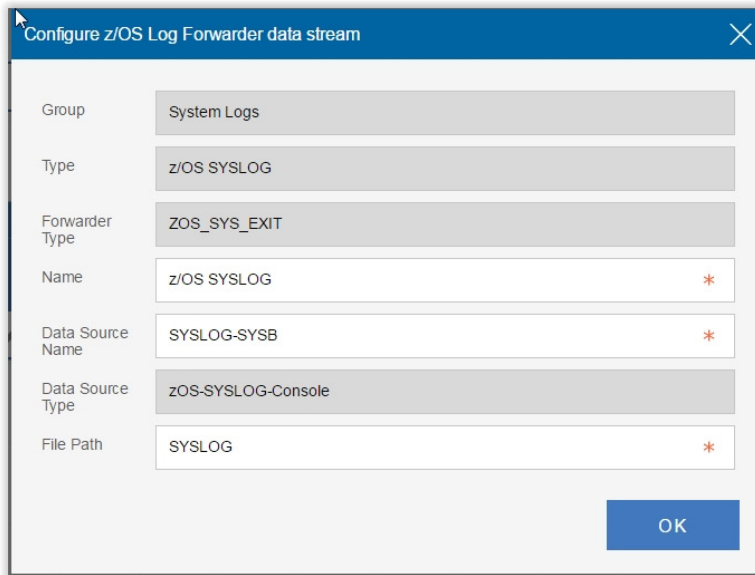
When editing a policy, a data stream is added by clicking on the + Data Stream Icon in the Policy editor:

The screenshot shows the 'Policy Profile Edit' window. At the top, there's a breadcrumb 'Common Data Provider > Profile Policy' and a status bar with 'Policy file directory: /etc/cdpConfig' and 'APAR OA52199, Build 121'. The main title is 'Policy Profile Edit' with 'CANCEL', 'IMPORT', and 'SAVE' buttons. Below, there's a 'Policy Name' field with 'testpolicy' and a 'Policy Description' text area. A 'Global Properties' section has tabs for 'SYSTEM', 'z/OS LOG FORWARDER', 'SDE', and 'SCHEDULES'. At the bottom, a grid of icons is visible, with a '+ DATA STREAM' icon highlighted.

On the subsequent window the z/OS SYSLOG data stream was selected, which captures data from the Common Data Provider Log Forwarder, as well as the option to transform to the UTF-8 code page:

The screenshot shows the 'Select data stream' dialog box. It has a search bar at the top. The main content is a tree view with categories: 'Log Data', 'Application Logs', 'Generic Feeds', 'SMF Data', 'Base Streams', and 'IOAz'. Under 'Log Data', there's a sub-category 'System Logs' with a table of data streams. The table has columns 'Name' and 'Tags'. The 'z/OS SYSLOG' stream is selected. At the bottom, there's a checkbox 'Transcribe selected streams to' with a dropdown set to 'UTF-8'. There are 'CANCEL' and 'SELECT' buttons.

Name	Tags
<input checked="" type="checkbox"/> z/OS SYSLOG	SYSLOG
<input type="checkbox"/> z/OS SYSLOG from OPERLOG	SYSLOG,OPERLOG,Split
<input type="checkbox"/> USS Syslogd Admin	Syslogd
<input type="checkbox"/> USS Syslogd Debug	Syslogd
<input type="checkbox"/> USS Syslogd Error	Syslogd




Configure z/OS Log Forwarder data stream

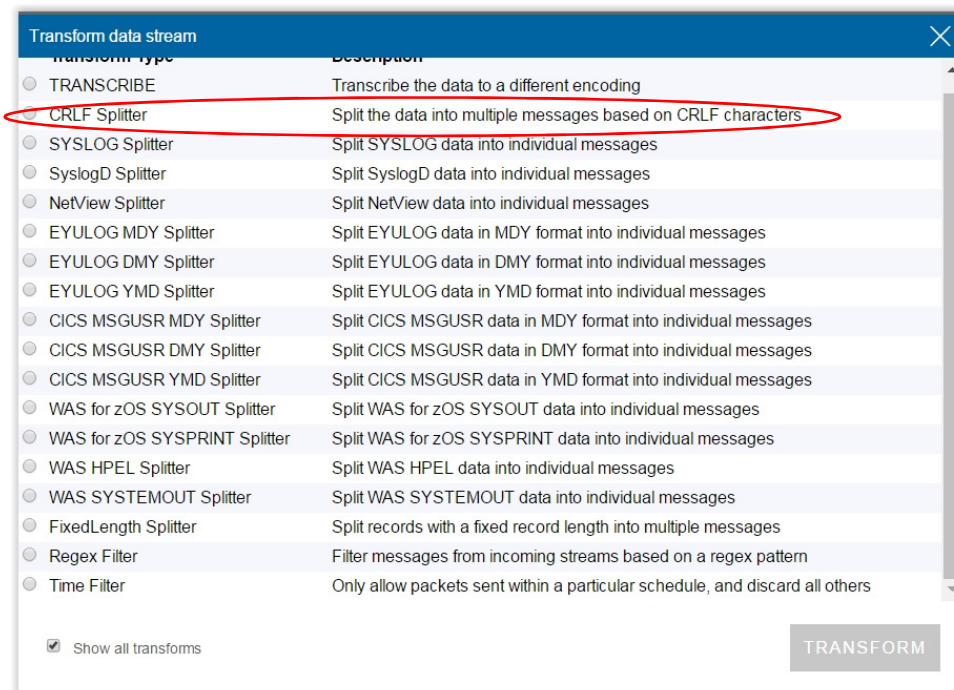
Group	System Logs
Type	z/OS SYSLOG
Forwarder Type	ZOS_SYS_EXIT
Name	z/OS SYSLOG *
Data Source Name	SYSLOG-SYSB *
Data Source Type	zOS-SYSLOG-Console
File Path	SYSLOG *

OK

These selections generate both the Data Stream and UTF-8 Transform icons in the workflow.

- Transform – Split

The Split transform is required to split the data so that each message is sent as a separate record. In the policy editor it is created by clicking on the  icon on the UTF-8 transform icon, which creates a new transform placed after the UTF-8 transform in the flow. For this example the SYSLOG Splitter option is used:




Transform data stream

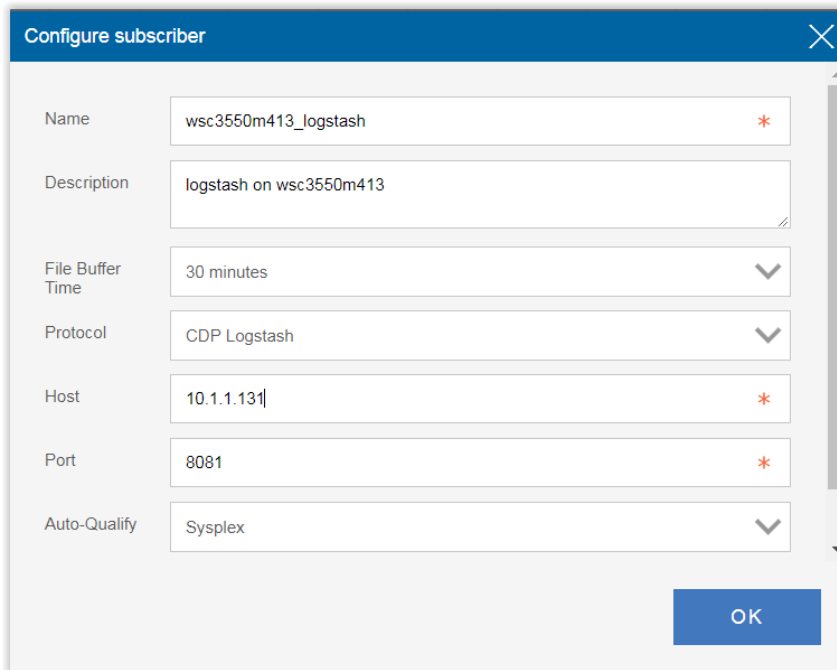
Transform type	Description
<input type="radio"/> TRANSCRIBE	Transcribe the data to a different encoding
<input checked="" type="radio"/> CRLF Splitter	Split the data into multiple messages based on CRLF characters
<input type="radio"/> SYSLOG Splitter	Split SYSLOG data into individual messages
<input type="radio"/> SyslogD Splitter	Split SyslogD data into individual messages
<input type="radio"/> NetView Splitter	Split NetView data into individual messages
<input type="radio"/> EYULOG MDY Splitter	Split EYULOG data in MDY format into individual messages
<input type="radio"/> EYULOG DMY Splitter	Split EYULOG data in DMY format into individual messages
<input type="radio"/> EYULOG YMD Splitter	Split EYULOG data in YMD format into individual messages
<input type="radio"/> CICS MSGUSR MDY Splitter	Split CICS MSGUSR data in MDY format into individual messages
<input type="radio"/> CICS MSGUSR DMY Splitter	Split CICS MSGUSR data in DMY format into individual messages
<input type="radio"/> CICS MSGUSR YMD Splitter	Split CICS MSGUSR data in YMD format into individual messages
<input type="radio"/> WAS for zOS SYSOUT Splitter	Split WAS for zOS SYSOUT data into individual messages
<input type="radio"/> WAS for zOS SYSPRINT Splitter	Split WAS for zOS SYSPRINT data into individual messages
<input type="radio"/> WAS HPEL Splitter	Split WAS HPEL data into individual messages
<input type="radio"/> WAS SYSTEMOUT Splitter	Split WAS SYSTEMOUT data into individual messages
<input type="radio"/> FixedLength Splitter	Split records with a fixed record length into multiple messages
<input type="radio"/> Regex Filter	Filter messages from incoming streams based on a regex pattern
<input type="radio"/> Time Filter	Only allow packets sent within a particular schedule, and discard all others

☒ Show all transforms

TRANSFORM

- Subscriber – wsc3350m143_logstash

The subscriber icon is created by clicking on the  icon in the Splitter transform and selecting Add Subscriber. The subscriber can then be configured:



The image shows a 'Configure subscriber' dialog box with the following fields and values:

Field	Value	Required
Name	wsc3550m413_logstash	*
Description	logstash on wsc3550m413	
File Buffer Time	30 minutes	
Protocol	CDP Logstash	
Host	10.1.1.131	*
Port	8081	*
Auto-Qualify	Sysplex	

An 'OK' button is located at the bottom right of the dialog.

- The Protocol attribute must be CDP Logstash (or CDP Logstash SSL, if using SSL).
- The Host attribute must be the hostname or IP address where the target Logstash instance is running.
- The Port attribute must match the port defined in the B_CDPz_Input.1sh file used in the Logstash configuration.

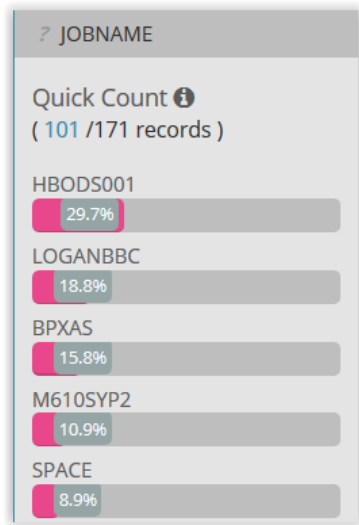
After the flow is defined in the policy, the Common Data Provider Data Streamer must be restarted to pick up the additions.

Results

In this example, when all of these components are running, z/OS SYSLOG messages will appear in Kibana.

Each record is in Comma Separated Value (CSV) format. The first seven fields of each record identify:

- Sysplex Name
- System Name
- Hostname
- Path (SYSLOG for a SYSLOG data stream source)
- Source Type (zOS-SYSLOG-CONSOLE for a SYSLOG data stream source)



Much more can be done with searching (such additional search conditions using field values and times intervals), build dashboards, etc. that are beyond the scope of this paper but documented in the Kibana documentation. The key is that once the z/OS SYSLOG data is available to the ELK stack, it can be treated just like any other data that comes into ELK.

Summary

While this example implementation is simple, it can be easily built upon to send additional data sources to ELK. The data rates should be monitored, as it can be desirable to set up several Logstash instances to receive different sets of data from CDPz, for performance purposes.

For further information on the Common Data Provider for z Systems, the following information sources are available:

- IBM Common Data Provider 1.1 Knowledge Center

https://www.ibm.com/support/knowledgecenter/SSGE3R_1.1.0/com.ibm.cdpz.doc/welcome.html

Contains links to program directories (SMP/E installation) and Users Guide (product setup, customization, and integrations).

- IBM Common Data Provider on IBM DeveloperWorks

<http://ibm.biz/CDPzWiki>

Contains FAQs, documentations, and product service information.