Progress®



Unified Logging for PAS for OpenEdge

OpenEdge®

Table of Contents

Copyright....5
Preface....7

Learn About Unified Logging....9
Prerequisites for unified logging in a container....9
Architecture for unified logging....10

Enable Unified Logging....11

PAS for OpenEdge Log Files....13
Common fields for logs....14
Structure of PAS for OpenEdge log files....14
pasoe_catalina_log....15
pasoe_localhost_log....15
pasoe_localhost_access_log....15
pasoe_application_log....16
pasoe_agent_log....16
pasoe_authn_log....16
pasoe_authz_log....17
start_server_log....17
fluent_bit_log....17

Fluent Bit Configuration Files....19
Customize Fluent Bit....20
Customize fields using MDC....21

Copyright

© 2021 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, DataRPM, Defrag This, Deliver More Than Expected, DevReach (and design), Icenium, Inspec, Ipswitch, iMacros, Kendo UI, Kinvey, MessageWay, MOVEit, NativeChat, NativeScript, OpenEdge, Powered by Chef, Powered by Progress, Progress, Progress Software Developers Network, SequeLink, Sitefinity (and Design), Sitefinity, Sitefinity (and design), SpeedScript, Stylus Studio, Stylized Design (Arrow/3D Box logo), Styleized Design (C Chef logo), Stylized Design of Samurai, TeamPulse, Telerik, Telerik (and design), Test Studio, WebSpeed, WhatsConfigured, WhatsConnected, WhatsUp, and WS_FTP are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries.

Analytics360, AppServer, BusinessEdge, Chef Automate, Chef Compliance, Chef Desktop, Chef Habitat, Chef WorkStation, Corticon.js, Corticon Rules, Data Access, DataDirect Autonomous REST Connector, DataDirect Spy, DevCraft, Fiddler, Fiddler Everywhere, FiddlerCap, FiddlerCore, FiddlerScript, Hybrid Data Pipeline, iMail, JustAssembly, JustDecompile, JustMock, KendoReact, NativeScript Sidekick, OpenAccess, PASOE, Pro2, ProDataSet, Progress Results, Progress Software, ProVision, PSE Pro, Push Jobs, SafeSpaceVR, Sitefinity Cloud, Sitefinity CMS, Sitefinity Digital Experience Cloud, Sitefinity Feather, Sitefinity Insight, Sitefinity Thunder, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, Supermarket, SupportLink, Unite UX, and WebClient are trademarks or service marks of Progress Software Corporation and/or its subsidiaries or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

July 2021

Product version: Progress OpenEdge 12.4

Preface

Purpose

This guide describes the unified logging concepts, procedures, and configurations that enable you to set up a unified logging solution with the PAS for OpenEdge container.

Audience

This guide is designed as a reference for developers that are working with the PAS for OpenEdge container.

Documentation conventions

See Documentation Conventions for an explanation of the terminology, format, and typographical conventions used throughout the OpenEdge content library.

1

Learn About Unified Logging

Unified logging benefits all types of deployment. Sorting through all available logs in a file system is inefficient. To alleviate this issue Progress recommends using a unified logging framework for better log monitoring. With unified logging, you get the opportunity to perform complex queries across all your log data in a single place and see different events side by side, establishing new insights and correlations.

Fluent Bit is included in the PAS for OpenEdge Docker container to collect log files.

Fluent Bit is a log processor that helps collect logs from multiple sources and then sends all the logs to various destinations. To know more about the process, see Fluent Bit Configuration Files on page 19

For details, see the following topics:

- Prerequisites for unified logging in a container
- Architecture for unified logging

Prerequisites for unified logging in a container

The following components are required for unified logging with the PAS for OpenEdge Docker container:

- A Docker environment
- A PAS for OpenEdge Docker container

Architecture for unified logging

With unified logging, you can collect and correlate logs across multiple sources, and display the relevant information in a simple interface.

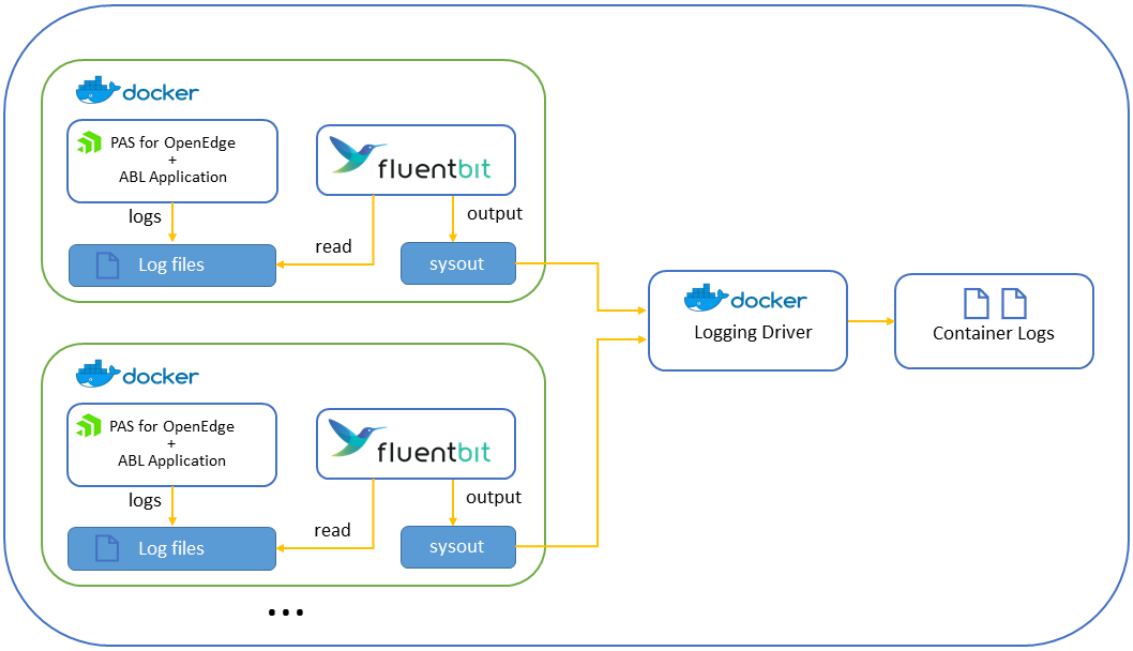
The following describes the architecture for PAS for OpenEdge and shows you how it works.

1. Fluent Bit collects the logs from PAS for OpenEdge Docker containers and enhances them to create a meaningful structure and adds metadata to the logs.
2. The collected logs are pushed to standard output (STDOUT).

   Note: STDOUT is one of the I/O streams. It usually refers to any command's normal output.

3. The standard output logs are captured as container logs based on the default logging driver of the Docker daemon or based on a different logging driver that is configured explicitly.

The following figure illustrates the Progress recommended approach for unified logging:



Note: It is one of the best practices of Docker Logging Design to stream logs to STDOUT from the container. For more information, see Logging Best Practices.

For more information about Fluent Bit, see Fluent Bit documentation.

2

Enable Unified Logging

To enable unified logging, Fluent Bit binaries and configurations are provided with the baseline PAS for OpenEdge container. The container image consists of the startup scripts, for starting and deploying applications in the `/deploy/scripts` directory. The startup script (`startServer.sh`) uses a flag to enable Fluent Bit.

To enable Fluent Bit logging, configure the following environment variables in the container:

- `FLUENTBIT_LOGGING`—Set this environment variable to `true` to enable log management with the Fluent Bit setup in the container. After this flag is set to `true`, and the container starts, the following command is executed:

  ```
  /usr/local/bin/fluent-bit -c /etc/fluent-bit/fluent-bit.conf
  ```

- `INSTANCE_NAME`—Set this environment variable to the name of the PAS for OpenEdge instance.
- `APP_NAME`—Set this environment variable to the logical name for the application to display in the logs.

Note: The `setup-fluentbit.sh` file is part of the startup script and is run after the application is deployed, before starting the Fluent Bit process. This script tailors the Fluent Bit configuration for PAS for OpenEdge application logs.

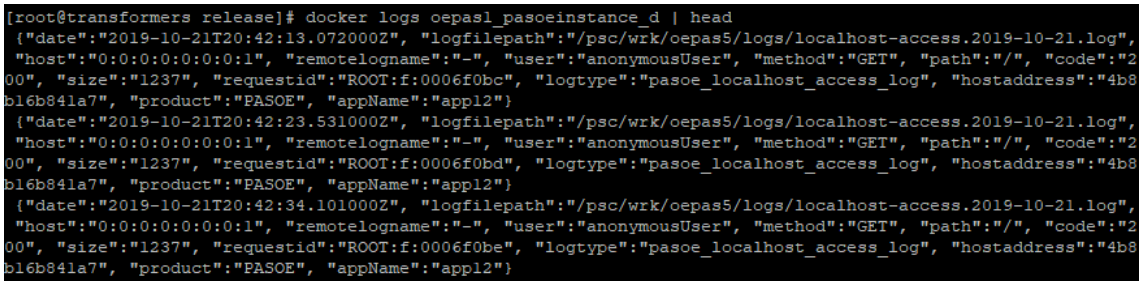
If you use the provided deployment scripts, Fluent Bit logging is enabled by default. For more information about the deployment process and how to deploy an application using these scripts, see "Deploy applications using scripts" in *Run PAS for OpenEdge in a Docker Container*.

The attributes you need for Fluent Bit logging are located in the `config.properties` file:

- `FLUENTBIT.LOGGING=true`
- `PAS.INSTANCE.NAME=`*`INSTANCE_NAME`*
- `APP.DOCKER.IMAGE.NAME=`*`APP_NAME`*

After you complete the configurations and Fluent Bit is started, you can view the logs by using the `docker logs` *`container_name`* command. The following image is a sample of how the logs are displayed.

Note: If you are using a Kubernetes environment, you can also view the logs using: `kubectl logs` *`pod_name`*.



3

PAS for OpenEdge Log Files

PAS for OpenEdge generates log files. For more information on PAS for OpenEdge log files, see "Log files" in *Learn about Progress Application Server for OpenEdge*.

Note: Session manager, host manager, and manager logs require manager functionality and are not available in PAS for OpenEdge.

Set up Fluent Bit logging

Use the following list of log files to set up your implementation of Fluent Bit logging:

- Tomcat
  - `catalina.out`
  - `catalina.`*`date`*`.log`
  - *`localhost.date`*`.log`
- ABL application
  - *`ABL-app-name.date`*`.log`
  - *`ABL-app-name`*`.agent.`*`date`*`.log`
- PAS for OpenEdge Authentication/Authorization Audit Trails (release 11.7.4+ and 12.0.0+)
  - *`ABL-app-name`*`_authn.`*`date`*`.log`
  - *`ABL-app-name`*`_authz.`*`date`*`.log`

Note: The *ABL-app-name*_authn.*date*.log and *ABL-app-name*_authz.*date*.log files are not generated by default. They are created when the relevant events of enabling authentication and authorization logs, are triggered.

The date and time in the PAS for OpenEdge log files are recorded in the UTC +0 format.

For details, see the following topics:

- Common fields for logs
- Structure of PAS for OpenEdge log files

Common fields for logs

The following are the common fields for all the log types of PAS for OpenEdge.

Log structure	Example
hostaddress	*HOSTNAME*
product	PASOE
appName	*APP_NAME*

Structure of PAS for OpenEdge log files

The following are the log files for PAS for OpenEdge is as follows:

- pasoe_catalina_log on page 15
- pasoe_localhost_log on page 15
- pasoe_localhost_access_log on page 15
- pasoe_application_log on page 16
- pasoe_agent_log on page 16
- pasoe_authn_log on page 16
- pasoe_authz_log on page 17
- start_server_log on page 17
- fluent_bit_log on page 17

pasoe_catalina_log

Log structure	Example
date verbosity [thread] loggername - messsage	2019-06-11T08:01:44.447Z INFO [main] org.apache.coyote.AbstractProtocol.init - Initializing ProtocolHandler ["https-jsse-nio-8811"]
Additional fields	
logtype	pasoe_catalina_log
logfilepath	/psc/wrk/*instance name*/logs/catalina.out or /psc/wrk/*instance name*/logs/catalina.*date*.log

pasoe_localhost_log

Log structure	Example
date verbosity [host_info] log_entry - messsage	2019-06-11T08:01:46.136Z INFO [Catalina-startStop-1] o.a.catalina.core.ApplicationContext.log - No Spring WebApplicationInitializer types detected on classpath
Additional fields	
logtype	pasoe_localhost_log
logfilepath	/psc/wrk/*instance name*/logs/localhost.*date*.log

pasoe_localhost_access_log

Log structure	Example
host remotelogname user [date] "method path HTTP/1.1" *code size requestid*	10.244.1.1 - anonymousUser [2019-06-11T08:02:24.708Z] "GET / HTTP/1.1" 200 1234 ROOT:f:00000001
Additional fields	
logtype	pasoe_localhost_access_log
logfilepath	/psc/wrk/*instance name*/logs/localhost-access.*date*.log

pasoe_application_log

Log structure	Example
date verbosity thread requestid loggername - message	2019-06-11T08:01:49.147Z WARN [Catalina-startStop-1] - c.p.a.s.s.OERequestHeaderAuthenticationFilter - Both key and registryFile specified. registryFile will be considered for validating C-P!!!
Additional fields	
logtype	pasoe_application_log
logfilepath	/psc/wrk/*instance name*/logs/*ablapp_name*.*date*.log

pasoe_agent_log

Log structure	Example
date pid thread loglevel executionenvironment requestid logentrytype message	2019-06-11T10:00:08.890+0000 015282 015286 1 AS-Aux-0 mtapsv:-:? MSAS Agent Starting Up -- Progress OpenEdge Release 12.0 build 1496.
Additional fields	
logtype	pasoe_agent_log
logfilepath	/psc/wrk/*instance name*/logs/*ablapp*.agent.*date*.log

pasoe_authn_log

Log structure	Example
date instance requestid eventstatus loginprincipal - failreason -faildetails	2019-06-11T10:14:46.096Z greet120 greet120:w:00000002 failure sadfkj - bad account secret - Bad credentials
Additional fields	
logtype	pasoe_authn_log
logfilepath	/psc/wrk/*instance name*/logs/*ablapp*_authn.*date*.log

pasoe_authz_log

Log structure	Example
date instance requestid eventstatus authzuser - [*failobject*] - [*failattrs*] - *failreason*	2019-06-11T10:14:36.653Z greet120 greet120:w:00000001 denied anonymousUser [/web/greet120] [hasAnyRole('ROLE_PSCUser')] - Access is denied
Additional fields	
logtype	pasoe_authz_log
logfilepath	/psc/wrk/*instance name*/logs/*ablapp*_authz.*date*.log

start_server_log

Log structure	Example
message	[exec] info: executing create Tomcat instance command
Additional fields	
logtype	start_server_log
logfilepath	/psc/wrk/logs/start-server.log

fluent_bit_log

Log structure	Example
message	^[[1m[^[[0m2019/06/11 10:13:29^[[1m]^[[0m [^[[92m info^[[0m] [engine] started (pid=15265)
Additional fields	
logtype	fluent_bit_log
logfilepath	/psc/wrk/logs/fluent-bit.log

4

Fluent Bit Configuration Files

PAS for OpenEdge provides option to customize existing logs or a PAS for OpenEdge application can create a new log file according to the business requirements. To get the advantage of unified logging, modify the Fluent Bit configurations to structure the logs.

The default configuration files for Fluent Bit are located in `/etc/fluent-bit/` directory inside the Docker container:

Configuration files	Descriptions
fluent-bit.conf	Runs Fluent Bit, the main configuration file. This file includes all the other configuration files.
fluent-bit-service.conf	Defines the Service plugin configurations.
fluent-bit-output.conf	Defines the Output plugin configurations.
io_agent.conf	Defines the input configurations for PAS for OpenEdge agent logs.
io_authn.conf	Defines the input configurations for PAS for OpenEdge authentication logs.
io_authz.conf	Defines the input configurations for PAS for OpenEdge authorization logs.
io_catalina.conf	Defines the input configurations for PAS for OpenEdge catalina logs.
io_localhost.conf	Defines the input configurations for PAS for OpenEdge localhost logs.
io_localhost_access.conf	Defines the input configurations for PAS for OpenEdge localhost access logs.
io_pasoe_application_*APP_NAME*.conf	Defines the input configurations for PAS for OpenEdge application logs.
io_fluent-bit.conf	Defines the input configurations for the running Fluent Bit verbose logs.
io_start-server.conf	Defines the input configurations for the Start Server script verbose logs.
parser_pasoe_agent.conf	Defines the parsers used by the input configurations.
parsers_custom.conf	Defines the custom parsers used by the user-defined input configurations.

For details, see the following topics:

- Customize Fluent Bit
- Customize fields using MDC

Customize Fluent Bit

You can customize Fluent Bit by making changes to the configuration files located in the `/etc/fluent-bit/` directory. For example, you can do the following:

- To change the Output plugin, by providing a custom configuration in `fluent-bit-output.conf` file.
- To change the Service plugin, by providing a custom configuration in `fluent-bit-service.conf` file.
- To change the log patterns, define a custom parser inside the `parsers_custom.conf` file and then modify the corresponding `log-file` configuration to point to the custom parser.
- If the PAS for OpenEdge application is generating additional log files, stream it to the STDOUT. To do that, add a new Input Fluent Bit configuration for that log file at `/etc/fluent-bit/` and include that configuration in the `fluent-bit.conf` file.

  For example, there is an extra log file at `/psc/wrk/logs/mylog.log`.

  Create an input configuration file, `io_mylog.conf`, with following content:

```
[INPUT]
    Name   tail
    path   /psc/wrk/logs/mylog.log
    Path_Key   logfilepath
    Tag    mylog_tag
    Parser mylog_single_line

[FILTER]
    Name record_modifier
    Match mylog_tag
    record logtype mylog_log
```

  Define the `mylog_single_line` in the `parsers_custom.conf` file, as shown:

```
[PARSER]
    Name    mylog_single_line
    Format  regex
    Regex   /regex/
    Time_Key    date
    Time_Format %Y/%m/%d %H:%M:%S
```

  Then, modify the `fluent-bit.conf` file to include this configuration:

```
@INCLUDE io_mylog.conf
```

Note: If you customize the default logging configuration of PAS for OpenEdge, change the Fluent Bit configurations.

Using the deployment scripts, you can add the changed configuration files or provide the complete set of configuration files in the `./deploy/conf/logging` folder.

Note: You can use these configurations in a PAS for OpenEdge, also and the output can be streamed to Elasticsearch or a similar service instead of STDOUT.

Customize fields using MDC

The Mapped Diagnostic Context (MDC) support is included in PAS for OpenEdge, so that you can add, remove, and rearrange fields to conform to the log file and scanning standards of your company, and to meet any company policies.

The Fluent Bit configuration, which is part of the PAS for OpenEdge image, extracts metadata from the log message from various log files and enriches the log message sent to log processing systems. MDC fields are also extracted and added as metadata to the log message, where applicable. For example:

```
{"@timestamp":"2019-01-16T12:08:04.880Z",
"date":"2019-01-16T12:08:04.880Z",
"instance":"oepas1",
"requestid":"Sports:f:00000002",
"eventstatus":"success",
"loginprincipal":"restuser",
"failreason":"",
"faildetails":"",
"hostaddress":"af41815a9a50",
"product":"PASOE",
"appName":"pasoe-basic-sample"}
```

In the example, the MDC request ID is extracted too.

Note: If changes are made to the MDC logging configuration, then update the Fluent Bit configurations.

For more information about the support for MDC, see "Configure logging in PAS for OpenEdge" in *Manage Progress Application Server (PAS) for OpenEdge*.