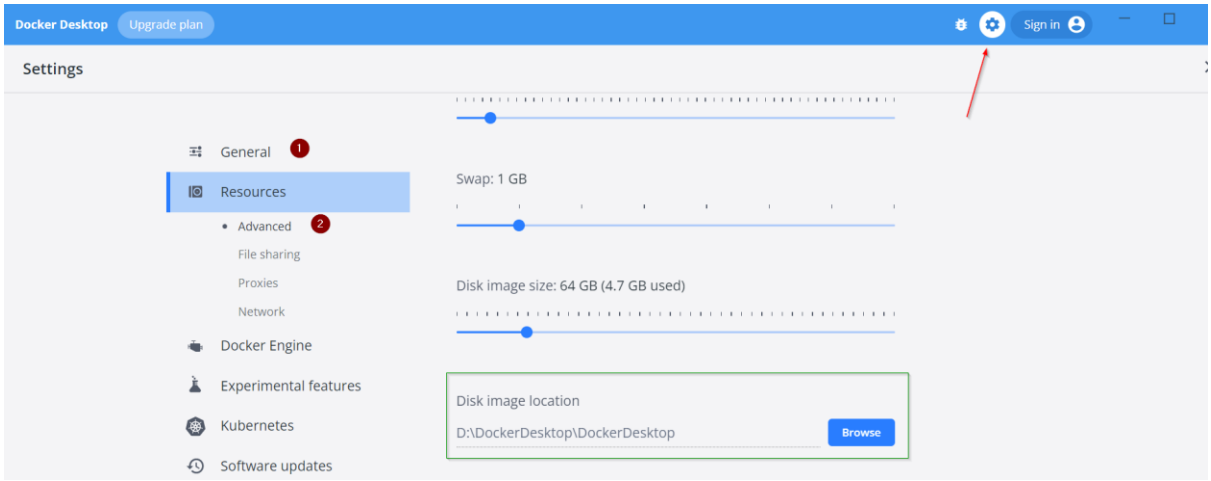
Prerequisites

Elasticsearch & Kibana should be up and running (in Docker container or as an application) on default ports (9200 and 5601 respectively). Token that was generated after the first Elastic startup or generated manually with elasticsearch-create-enrollment-token utility, is added to Kibana. I used this guide:
https://www.elastic.co/guide/en/kibana/current/docker.html
NOTE! Docker images take a lot of C:\ drive disk space. Looks like it is impossible to change Docker installation drive! But it is possible to change a folder where images are saved in Advanced (2) options. Checkbox Use the WSL 2 based engine on General (1) should be absent, and Hyper-V is turned on (for Windows)



Filebeat installation and configuration

1. Install Filebeat as a service on a PC where log files are collected according to manual https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-installation-configuration.html
   In 5th step, I used a command `.\filebeat.exe run` to start Filebeat. This command has many useful features, like `test`. To see all of them, type `-help`:

```
PS C:\Program Files\Filebeat> .\filebeat.exe --help
Usage:
  filebeat [flags]
  filebeat [command]

Available Commands:
  export      Export current config or index template
  generate    Generate Filebeat modules, filesets and fields.yml
  help        Help about any command
  keystore    Manage secrets keystore
  modules     Manage configured modules
  run         Run filebeat
  setup       Setup index template, dashboards and ML jobs
  test        Test config
  version     Show current version info

Flags:
  -E, --E setting=value              Configuration overwrite
  -M, --M setting=value              Module configuration overwrite
  -N, --N                            Disable actual publishing for testing
  -c, --c string                     Configuration file, relative to path.config (default "filebeat.yml")
      --cpuprofile string            Write cpu profile to file
  -d, --d string                     Enable certain debug selectors
  -e, --e                            Log to stderr and disable syslog/file output
      --environment environmentVar   set environment being ran in (default default)
  -h, --help                         help for filebeat
      --httpprof string              Start pprof http server
      --memprofile string            Write memory profile to this file
      --modules string               List of enabled modules (comma separated)
      --once                         Run filebeat only once until all harvesters reach EOF
      --path.config string           Configuration path
      --path.data string             Data path
      --path.home string             Home path
      --path.logs string             Logs path
      --strict.perms                 Strict permission checking on config files (default true)
  -v, --v                            Log at INFO level

Use "filebeat [command] --help" for more information about a command.
PS C:\Program Files\Filebeat> .\filebeat.exe run
```

2. Prepare configuration file. Mine was like this:

```
filebeat.inputs:
- type: filestream
  id: my-filestream-id
  enabled: true
  paths:
    - D:\DEV\demostock\logs\*.log

filebeat.config.modules:
  path: ${path.config}/modules.d/*.yml

  # Set to true to enable config reloading
  reload.enabled: false

setup.template.settings:
  index.number_of_shards: 1

setup.kibana:
  host: "localhost:5601"
  username: "elastic"
  password: "9w=8x+XZY5xS1_q0VirV"


output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["https://localhost:9200"]
  pipeline: parse_log
```

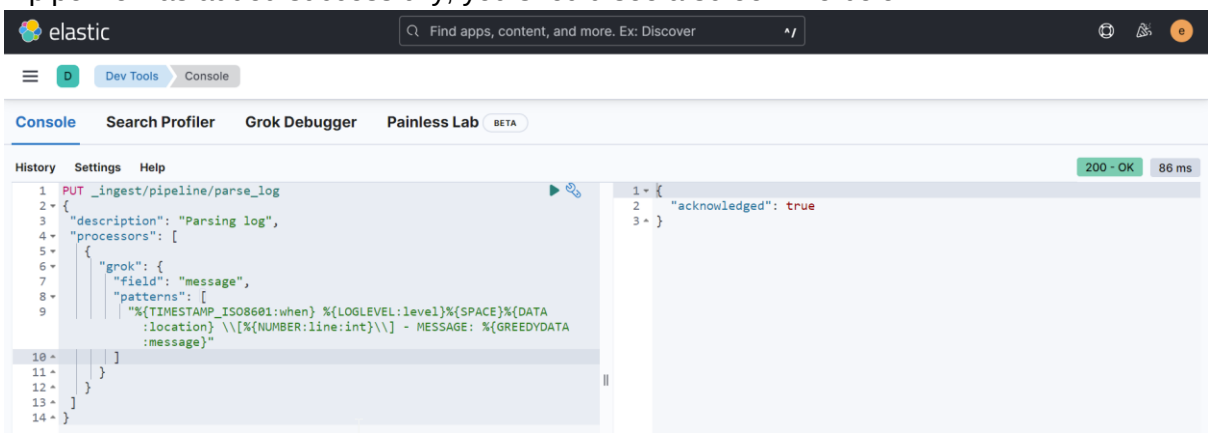
```
  username: "elastic"
  password: "9w=8x+XZY5xS1_q0VirV"
  ssl:
    enabled: true
    ca_trusted_fingerprint:
"b0d800b7d27de668a129b8d3b50e298947541fe61055e3d21707d72112503203"


processors:
  - add_host_metadata:
      when.not.contains.tags: forwarded
  - add_cloud_metadata: ~
  - add_docker_metadata: ~
  - add_kubernetes_metadata: ~
```

Logs parsing

To parse log messages, I added a line **`pipeline: parse_log`**
where parse_log is a pipeline that should be added in Elastic's Dev console:

```
PUT _ingest/pipeline/parse_log
{
 "description": "Parsing log",
 "processors": [
   {
     "grok": {
       "field": "message",
       "patterns": [
         "%{TIMESTAMP_ISO8601:when} %{LOGLEVEL:level}%{SPACE}%{DATA:location}
\\[%{NUMBER:line:int}\\] - MESSAGE: %{GREEDYDATA:message}"
       ]
     }
   }
 ]
}
```

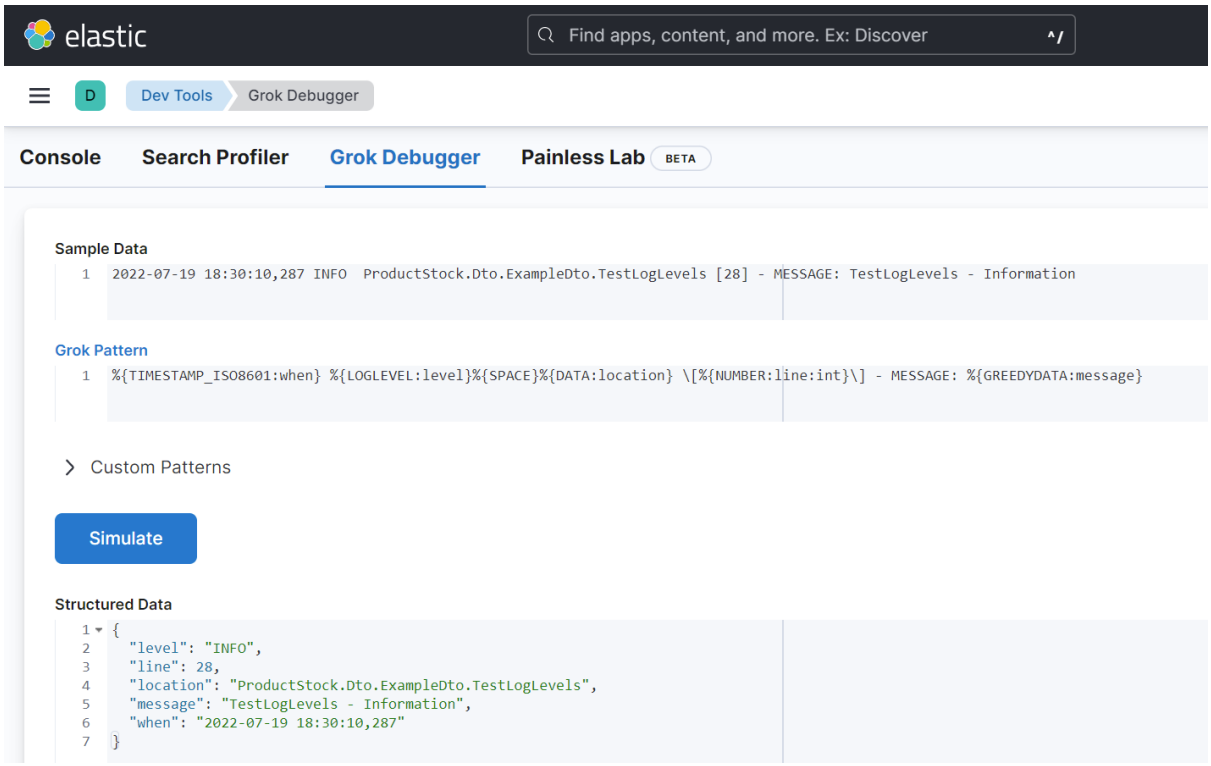
If pipeline was added successfully, you should see a screen like below:



Please read more for details:
https://www.elastic.co/guide/en/elasticsearch/reference/8.3/grok.html
https://www.elastic.co/guide/en/elasticsearch/reference/current/ingest.html#pipelines-for-beats
Here is a pattern list that can be used for parsing:

https://github.com/elastic/elasticsearch/blob/master/libs/grok/src/main/resources/patterns/legacy/grok-patterns

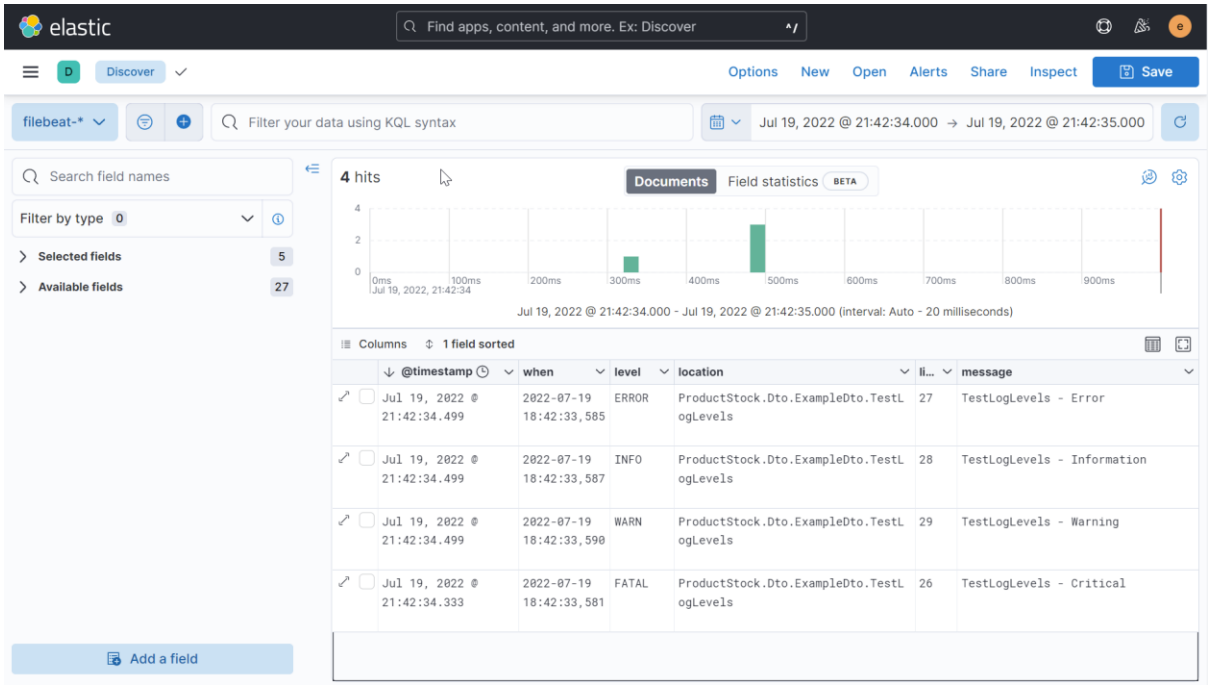
To check the pattern works, I used a Grok Debugger:



Having these logs:

2022-07-19 18:53:46,949 FATAL ProductStock.Dto.ExampleDto.TestLogLevels [26] - MESSAGE: TestLogLevels - Critical
2022-07-19 18:53:46,954 ERROR ProductStock.Dto.ExampleDto.TestLogLevels [27] - MESSAGE: TestLogLevels - Error
2022-07-19 18:53:46,957 INFO ProductStock.Dto.ExampleDto.TestLogLevels [28] - MESSAGE: TestLogLevels - Information
2022-07-19 18:53:46,960 WARN ProductStock.Dto.ExampleDto.TestLogLevels [29] - MESSAGE: TestLogLevels - Warning

I got a nice table:



Updates to process multiline messages and stack trace

log4net.config has filters to separate different level logs. You should add a `filter` node to do so. For messages above WARN, this configuration will print 4 last lines of stack trace starting from next line:

```
<log4net>
	<root>
		<level value="INFO" />
		<appender-ref ref="UpToWarnings" />
		<appender-ref ref="ExceptionsAndAbove" />
	</root>
	<appender name="UpToWarnings" type="log4net.Appender.RollingFileAppender">
		<filter type="log4net.Filter.LevelRangeFilter">
			<levelMax value="WARN" />
			<acceptOnMatch value="true" />
		</filter>
		<appendToFile value="true" />
		<file value="D:\DEV\demostock\logs\logfile" />
		<staticLogFileName value="false" />
		<rollingStyle value="Date" />
		<datePattern value="yyyyMMdd-HHmm.lo\g" />
		<layout type="log4net.Layout.PatternLayout">
			<conversionPattern value="[%utcdate] %-5level %logger.%method
[%line] - MESSAGE: %message%newline" />
		</layout>
	</appender>
	<appender name="ExceptionsAndAbove"
type="log4net.Appender.RollingFileAppender">
		<filter type="log4net.Filter.LevelRangeFilter">
			<levelMin value="ERROR" />
```

```
                    <acceptOnMatch value="true" />
                </filter>
                <appendToFile value="true" />
                <file value="D:\DEV\demostock\logs\logfile" />
                <staticLogFileName value="false" />
                <rollingStyle value="Date" />
                <datePattern value="yyyyMMdd-HHmm.err" />
                <layout type="log4net.Layout.PatternLayout">
                    <conversionPattern value="[%utcdate] %-5level %logger.%method
[%line] - MESSAGE: %message %newline CALLSTACK: %stacktrace{4}%newline" />
                </layout>
        </appender>
</log4net>
```

Filebeat converts multiline message into one line. Here are details

Because errors and information logs were divided into two files, one more file path was added into configuration. The updated filebeat.yml is:

```
filebeat.inputs:
- type: filestream
  id: my-filestream-id
  enabled: true
  paths:
    - D:\DEV\demostock\logs\*.log
    - D:\DEV\demostock\logs\*.err
  parsers:
    - multiline:
        type: pattern
        pattern: '^\['
        negate: true
        match: after
filebeat.config.modules:
  # Glob pattern for configuration loading
  path: ${path.config}/modules.d/*.yml

  # Set to true to enable config reloading
  reload.enabled: false
setup.template.settings:
  index.number_of_shards: 1
setup.kibana:
  host: "localhost:5601"
  username: "elastic"
  password: "9w=8x+XZY5xS1_q0VirV"
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["https://localhost:9200"]
  pipeline: parse_log
  username: "elastic"
  password: "9w=8x+XZY5xS1_q0VirV"
  ssl:
    enabled: true
    ca_trusted_fingerprint:
"b0d800b7d27de668a129b8d3b50e298947541fe61055e3d21707d72112503203"
processors:
  - add_host_metadata:
      when.not.contains.tags: forwarded
  - add_cloud_metadata: ~
  - add_docker_metadata: ~
  - add_kubernetes_metadata: ~
```

Updated pipeline is:

```
PUT _ingest/pipeline/parse_log
{
 "description": "Parsing log",
 "processors": [
   {
     "grok": {
       "field": "message",
       "patterns": [
         "\\[%{TIMESTAMP_ISO8601:when}\\] %{LOGLEVEL:level}%{SPACE}%{DATA:location}
\\[%{NUMBER:line:int}\\] -
MESSAGE:(?<message>.+?(?=\r\n|$))(%{SPACE}CALLSTACK:)?%{GREEDYDATA:stacktrace}"
       ]
     }
   }
 ]
}
```

[Please read more about custom patterns here]

Highlighted with a blue color expression should be interpreted as: put into `<message>` everything before line break `\r\n` or before end of the line `$`. Then optionally (everything inside `()?` is optional) find whitespaces and `CALLSTACK:` Put all the rest in stacktrace.

Having such .log and .err files:

[2022-07-20 19:18:17,936] INFO ProductStock.Dto.ExampleDto.MoveNext [0] - MESSAGE: TestLogLevels - Information
[2022-07-20 19:18:17,940] WARN ProductStock.Dto.ExampleDto.MoveNext [0] - MESSAGE: TestLogLevels - Warning

[2022-07-20 19:18:17,928] ERROR ProductStock.Dto.ExampleDto.MoveNext [29] - MESSAGE: TestLogLevels - Error
CALLSTACK:
System.Runtime.CompilerServices.AsyncTaskMethodBuilder`1+AsyncStateMachineBox`1.MoveNext > System.Threading.ExecutionContext.RunInternal > System.Runtime.CompilerServices.AsyncTaskMethodBuilder`1+AsyncStateMachineBox`1.ExecutionContextCallback > ProductStock.Api.Controllers.ExampleController+<TestLogLevels>d__3.MoveNext
[2022-07-20 19:18:17,933] FATAL ProductStock.Dto.ExampleDto.MoveNext [0] - MESSAGE: TestLogLevels - Critical
CALLSTACK:
System.Runtime.CompilerServices.AsyncTaskMethodBuilder`1+AsyncStateMachineBox`1.MoveNext > System.Threading.ExecutionContext.RunInternal > System.Runtime.CompilerServices.AsyncTaskMethodBuilder`1+AsyncStateMachineBox`1.ExecutionContextCallback > ProductStock.Api.Controllers.ExampleController+<TestLogLevels>d__3.MoveNext

Kibana parses like below (columns `line, when` and so on still exists, just are not shown in the table):