Ingest Pipeline Guide

Ingest Pipelines – Structured Log Parsing

Log Processor includes built-in OpenSearch ingest pipelines that automatically parse common log formats into structured fields at index time. Raw messages are preserved – parsed fields are added alongside them, enabling both full-text search and field-level filtering, aggregations, and dashboards.

How It Works

Each subscription stream can specify a pipeline field. When events from that stream are indexed into OpenSearch, the pipeline runs server-side and extracts structured fields from the message text. If parsing fails, the event is indexed unchanged – no data loss.

```
{
  "logGroupName": "/aws/lambda/my-app",
  "streams": [{
    "logStreamName": ".*",
    "index": "app",
    "pipeline": "lambda",
    "targets": ["opensearch", "datalake"]
  }]
}
```

Built-in Pipelines

Pipeline	Format	Extracted Fields
lambda	AWS Lambda runtime	log_timestamp, request_id, level, msg
json	JSON structured	All top-level keys merged into document
nginx	Nginx combined access log	client_ip, method, path, status, bytes, user_agent
apache	Apache combined access log	client_ip, method, path, status, bytes, user_agent
syslog	RFC 3164 syslog	syslog_timestamp, hostname, program, pid, msg
tomcat	Tomcat/Catalina	log_timestamp, level, thread, class, method, msg
spring	Spring Boot	log_timestamp, level, pid, thread, class, msg
vpc-flow	VPC Flow Logs	srcaddr, dstaddr, srcport, dstport, protocol, packets, bytes, action
alb	ALB access logs	client_ip, method, path, elb_status_code, target_processing_time, user_agent
api-gateway	API Gateway (JSON)	request_id + all JSON fields
rds-slow	RDS/MySQL slow query	log_timestamp, user, host, query_time, lock_time, rows_sent, rows_examined
eks	Parses the Kubernetes container log wrapper	timestamp stream logtag message

cloudfront	Parses CloudFront tab-separated access logs	client_ip, method, path, status, bytes_sent, bytes_received, time_taken, edge_location, edge_result_type, user_agent, ssl_protocol, ttfb, etc.
rds-postgres	Parses RDS PostgreSQL log format	log_timestamp, client_ip, client_port, user, database, pid, level, msg

Configuration

Set the pipeline field on any stream in the Subscription Editor or directly in subscriptions.json. The editor provides a dropdown with all built-in pipeline names and accepts custom names.

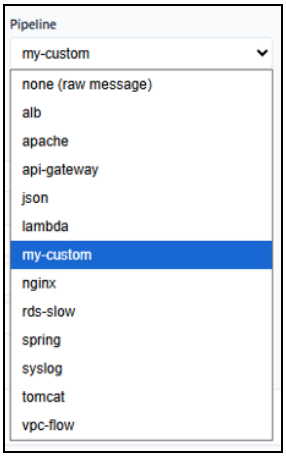
- Leave empty or omit for no parsing (raw message only)
- Select a built-in name for automatic parsing
- Enter a custom name to use a pipeline you created via the OpenSearch Dev Tools console

Custom Pipelines

Create your own pipelines via the OpenSearch API:

```
PUT _ingest/pipeline/my-custom
{
  "description": "Parse my app's log format",
  "processors": [{
    "grok": {
      "field": "message",
      "patterns": ["%{TIMESTAMP_ISO8601:ts} %{LOGLEVEL:level} %{GREEDYDATA:msg}"],
      "ignore_failure": true
    }
  }]
}
```

Upon adding a new pipeline, the editor will automatically include the new pipeline parser:



Note: If the pipeline doesn't exist at index time, events are indexed without parsing and a warning is logged.

Datalake Impact

Pipelines only affect OpenSearch indexing. Datalake (Athena) writes always store the raw message field. Use Athena's regexp_extract() or json_extract_scalar() for query-time parsing.

Feature Availability

- All tiers with OpenSearch (essential+): built-in pipelines available
- Custom pipelines: create via OpenSearch Dev Tools on any tier with OpenSearch
- Basic tier (datalake only): not applicable – no OpenSearch

Troubleshooting

- **Fields not appearing:** Verify the pipeline name matches exactly (case-sensitive). Check GET _ingest/pipeline/{name} exists.
- **Parse failures:** Events with unparseable messages are indexed with message only – no error, no data loss. Check for _pipeline_parse_error field to find JSON parse failures.
- **Invalid pipeline name:** If the pipeline doesn't exist, events are automatically retried without the pipeline. A warning appears in the Lambda logs and CloudWatch monitoring dashboard.