**What is a Watcher?**

A **Watcher** in Elastic Search is part of the **X-Pack** feature that allows you to monitor data and **trigger alerts** based on conditions. You can use it to notify you of anomalies, failures, thresholds being breached, etc.

**🧱 Basic Components of a Watch**

1. **Trigger**: When the watch runs (e.g., every 5 minutes)
2. **Input**: What data the watch will check (e.g., search query, HTTP, or none)
3. **Condition**: Logic that checks if the data meets certain criteria
4. **Actions**: What to do if the condition is true (e.g., send email, log, webhook)

**🧪 Example 1: Simple Watch with Search Input and Logging Action**

PUT \_watcher/watch/error\_log\_alert

{

"trigger": {

"schedule": {

"interval": "5m"

}

},

"input": {

"search": {

"request": {

"indices": ["logs-\*"],

"body": {

"query": {

"match": {

"log.level": "error"

}

}

}

}

}

},

"condition": {

"compare": {

"ctx.payload.hits.total.value": {

"gt": 0

}

}

},

"actions": {

"log\_error": {

"logging": {

"text": "Found {{ctx.payload.hits.total.value}} error logs!"

}

}

}

}

**⚙️ Example 2: Watch with http Input and email Action**

PUT \_watcher/watch/check\_api\_health

{

"trigger": {

"schedule": {

"interval": "1h"

}

},

"input": {

"http": {

"request": {

"method": "get",

"url": "https://your-api-url/health"

}

}

},

"condition": {

"compare": {

"ctx.payload.status\_code": {

"not\_eq": 200

}

}

},

"actions": {

"send\_email": {

"email": {

"to": "admin@example.com",

"subject": "API Health Check Failed",

"body": {

"text": "The API returned status {{ctx.payload.status\_code}}"

}

}

}

}

}

**🔁 Example 3: Watch with No Input and Just a Log Message**

PUT \_watcher/watch/static\_log\_message

{

"trigger": {

"schedule": {

"interval": "10m"

}

},

"input": {

"none": {}

},

"condition": {

"always": {}

},

"actions": {

"log\_action": {

"logging": {

"text": "This is a scheduled log message from Watcher."

}

}

}

}

**📩 Example 4: Watch with Webhook Action (for Slack or REST API)**

PUT \_watcher/watch/post\_to\_webhook

{

"trigger": {

"schedule": {

"interval": "15m"

}

},

"input": {

"search": {

"request": {

"indices": ["system-metrics-\*"],

"body": {

"query": {

"range": {

"cpu.usage": {

"gte": 90

}

}

}

}

}

}

},

"condition": {

"compare": {

"ctx.payload.hits.total.value": {

"gt": 0

}

}

},

"actions": {

"notify\_webhook": {

"webhook": {

"method": "POST",

"url": "https://your-webhook-url",

"body": {

"text": "⚠️ High CPU Usage Detected"

}

}

}

}

}

**🔍 Example 5: Watch Using Aggregation Input and Email**

PUT \_watcher/watch/log\_count\_alert

{

"trigger": {

"schedule": {

"interval": "5m"

}

},

"input": {

"search": {

"request": {

"indices": ["logs-\*"],

"body": {

"size": 0,

"aggs": {

"log\_levels": {

"terms": {

"field": "log.level.keyword"

}

}

}

}

}

}

},

"condition": {

"script": {

"source": """

return ctx.payload.aggregations.log\_levels.buckets.stream()

.anyMatch(bucket -> bucket.key == 'error' && bucket.doc\_count > 10);

"""

}

},

"actions": {

"email\_admin": {

"email": {

"to": "alerts@example.com",

"subject": "Too many error logs!",

"body": {

"text": "There are more than 10 error logs in the last 5 minutes."

}

}

}

}

}

**🧩 Bonus: Watch with Multiple Actions**

You can chain multiple actions together:

"actions": {

"log\_error": {

"logging": {

"text": "Error found"

}

},

"email\_admin": {

"email": {

"to": "admin@example.com",

"subject": "Alert Triggered",

"body": {

"text": "Check logs immediately."

}

}

}

}

**🛠 Teaching Tips**

* Start with a **simple logging action** before moving to **email/webhook**
* Use **Dev Tools** in Kibana to create and test watchers
* Show how to simulate a watcher:

bash

CopyEdit

POST \_watcher/watch/<watch\_id>/\_execute

* Emphasize security: email and webhooks should be secured

Cron Trigger

{

"trigger": {

"schedule": {

"cron": ["\*/10 \* \* \* \* ?"]

}

},

"checks": [],

"actions": [],

}