Achieving High Ovservability through Elastic Stack

Samidhya Sarker
Assistant Programmer
Bangladesh Computer Council

BANGLADESH COMPUTER COUNCIL

Presentation Outline

1. Observability
    1. Observability of Applications vs Observability of Systems
        1. Observability of Systems
        2. Observability of Applications
        3. Overlap and Differences
    2. Roles
    3. Elastic Architecture
        1. Ingestion of Logs
        2. Fleet Server Architecture
        3. Beats Architecture
2. Systems Monitoring
    1. SRE Performance Monitoring
    2. Custom Dashboard for VMs
        1. Single VMs
        2. Multiple VMs
4. Applications Performance Monitoring
    1. APM Tools
        1. 1. Counters
    2. Counters
    3. Profiling
    4. Tracing
    5. Log Ingestion
        1. Built in integrations
        2. Custom Logs
        3. Ingest Pipeline
            1. Processor
    6. Network Schematic

Observability

Observability refers to understanding a system through **observation**, and classifies the tools


Observability of Applications vs Observability of Systems

Observability of Systems

- Concentrates on monitoring and analyzing the **underlying infrastructure** (e.g., servers, networks, containers) that applications rely on. It includes:
  - **Key Metrics**: CPU, memory, disk I/O, network throughput, container health, etc.
  - **Events**: Logs related to system-level changes or failures (e.g., kernel panics, node crashes).
  - **Topology**: Relationships and dependencies between system components, such as container orchestration with Kubernetes.
  - **Primary Tools**: Infrastructure monitoring tools like Prometheus, Grafana, Nagios, or Splunk.
  - **Use Case**: Diagnosing issues like high server load, network latency, or disk space exhaustion.

Observability of Applications

- Focuses on monitoring and understanding the behavior, performance, and health of **software applications**.

  It centers around:

  - **Key Metrics**: Response times, throughput, error rates, request volumes, etc.
  - **Tracing**: End-to-end visibility of user interactions, such as tracing a request through APIs, services, and databases.
  - **Logs**: Application-specific logs for debugging, capturing exceptions, or auditing actions.
  - **Primary Tools**: Application Performance Monitoring (APM) solutions like New Relic, Dynatrace, or Datadog.
  - **Use Case**: Troubleshooting issues like slow API responses, memory leaks, or application-level bugs.

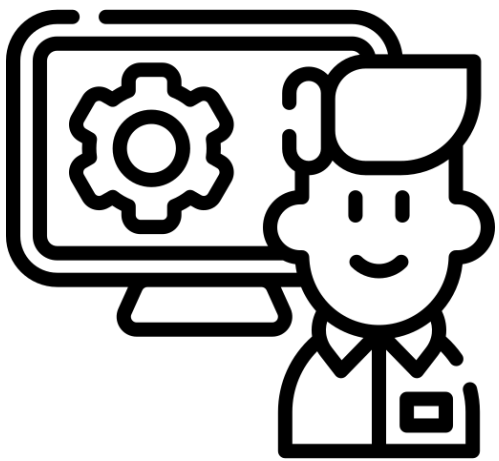
Overlap and Differences

- **Overlap**: Both share some concerns, such as logging and metrics, but differ in granularity and scope. They often feed into the same dashboards for holistic observability.
- **Differences**:
  - Applications observability is **user-centric**, focusing on service behavior and business logic.
  - Systems observability is **resource-centric**, focusing on the infrastructure and operational concerns.

To achieve comprehensive observability, combine both domains, ensuring both application performance and system health are monitored cohesively.

Roles

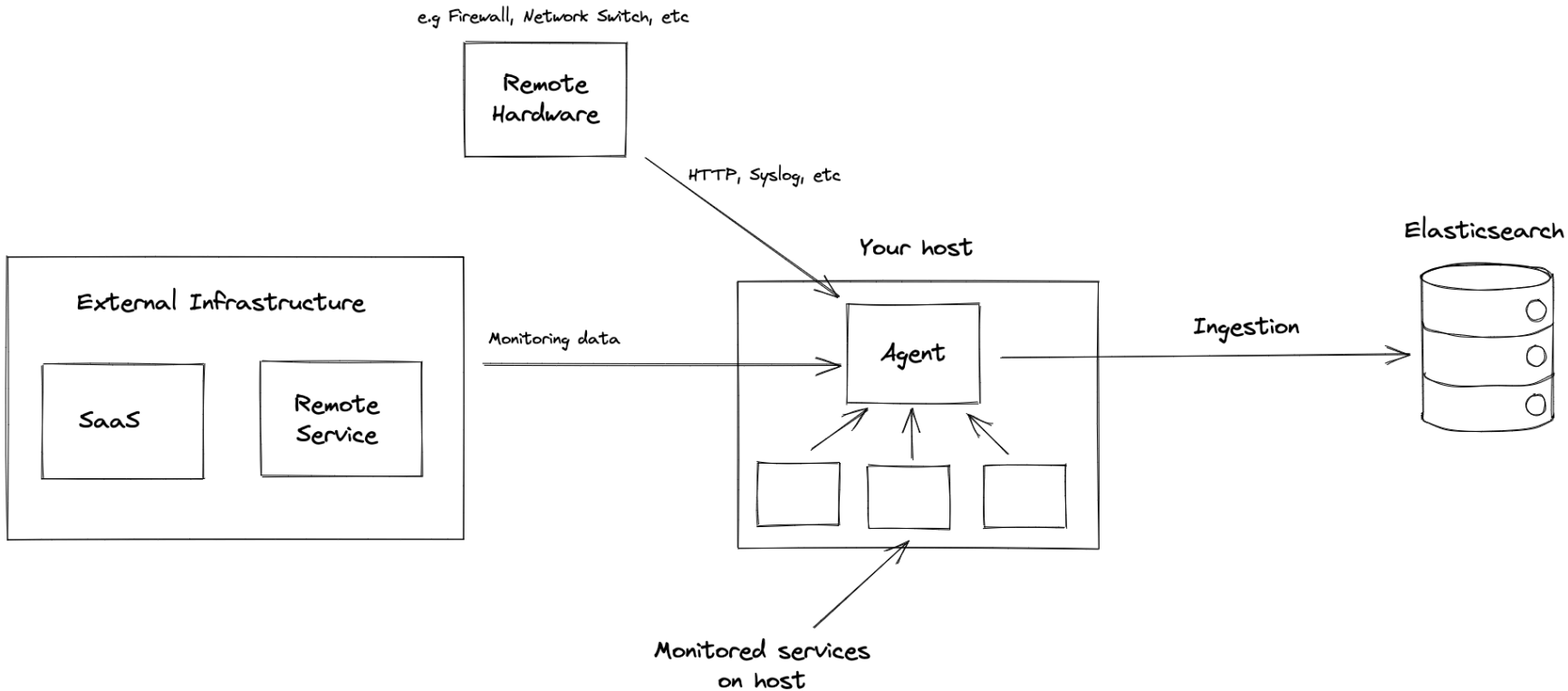
Sysadmin



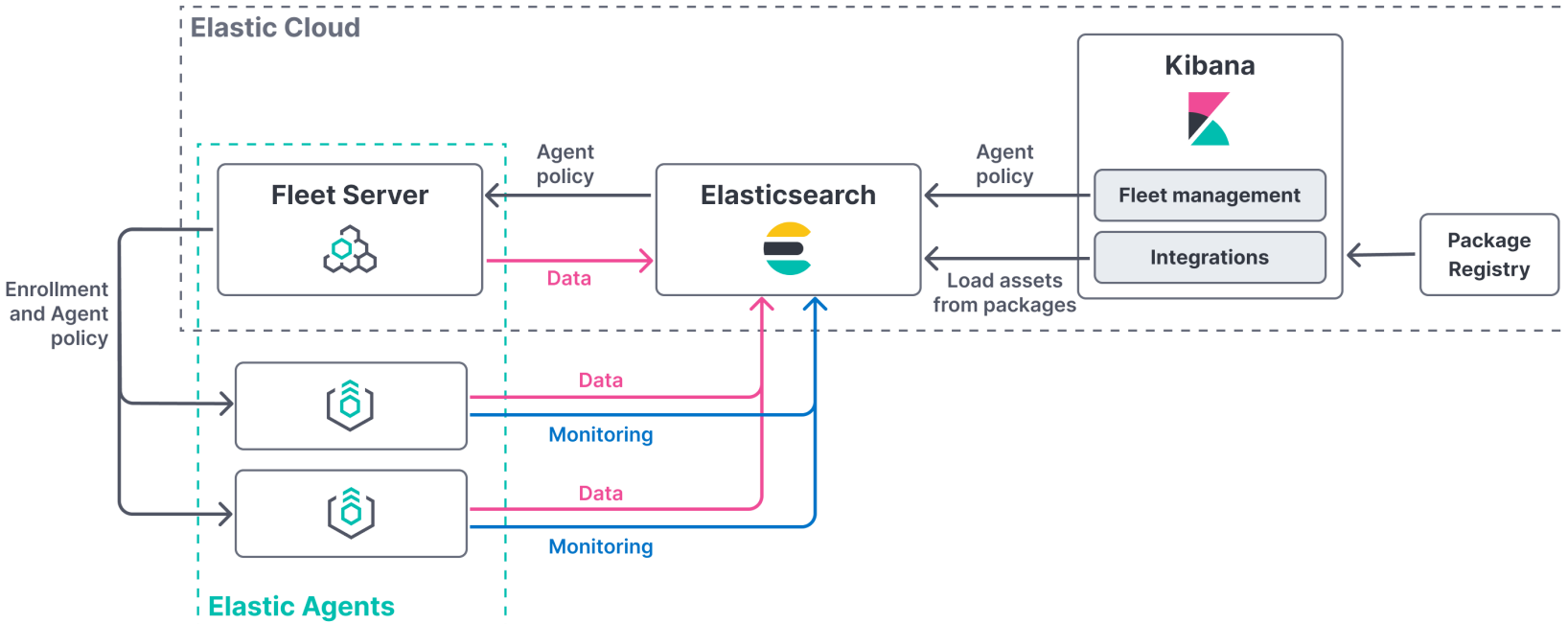
Developer

Elastic Architecture

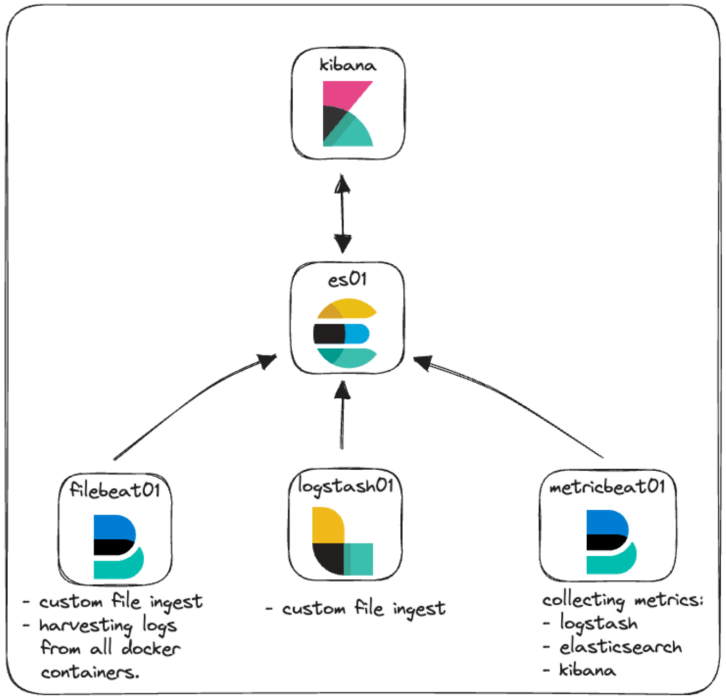
Ingestion of Logs



Fleet Server Architecture



Beats Architecture



BANGLADESH COMPUTER COUNCIL

Systems Monitoring

BANGLADESH COMPUTER COUNCIL

SRE Performance Monitoring

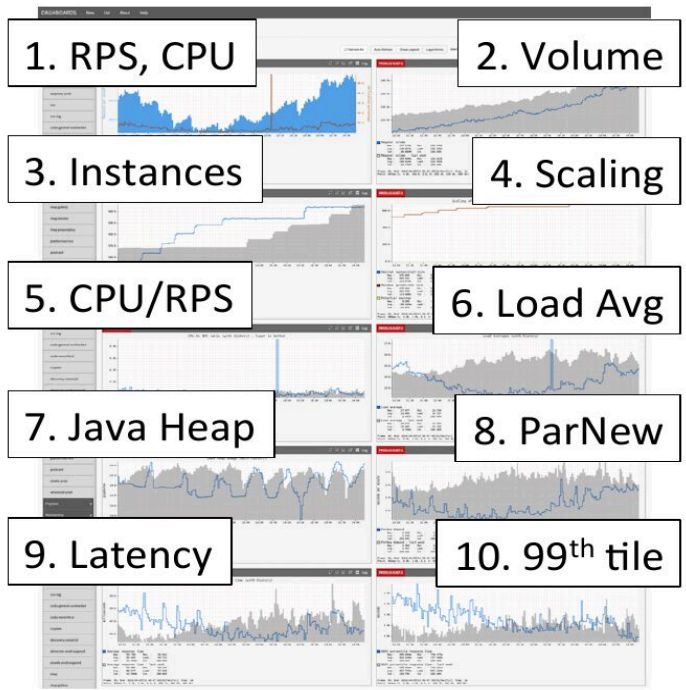
https://www.brendangregg.com/Slides/SREcon_2016_perf_checklists.pdf

Performance Checklists

per instance:

1. `uptime`
2. `dmesg -T | tail`
3. `vmstat 1`
4. `mpstat -P ALL 1`
5. `pidstat 1`
6. `iostat -xz 1`
7. `free -m`
8. `sar -n DEV 1`
9. `sar -n TCP,ETCP 1`
10. `top`

cloud wide:



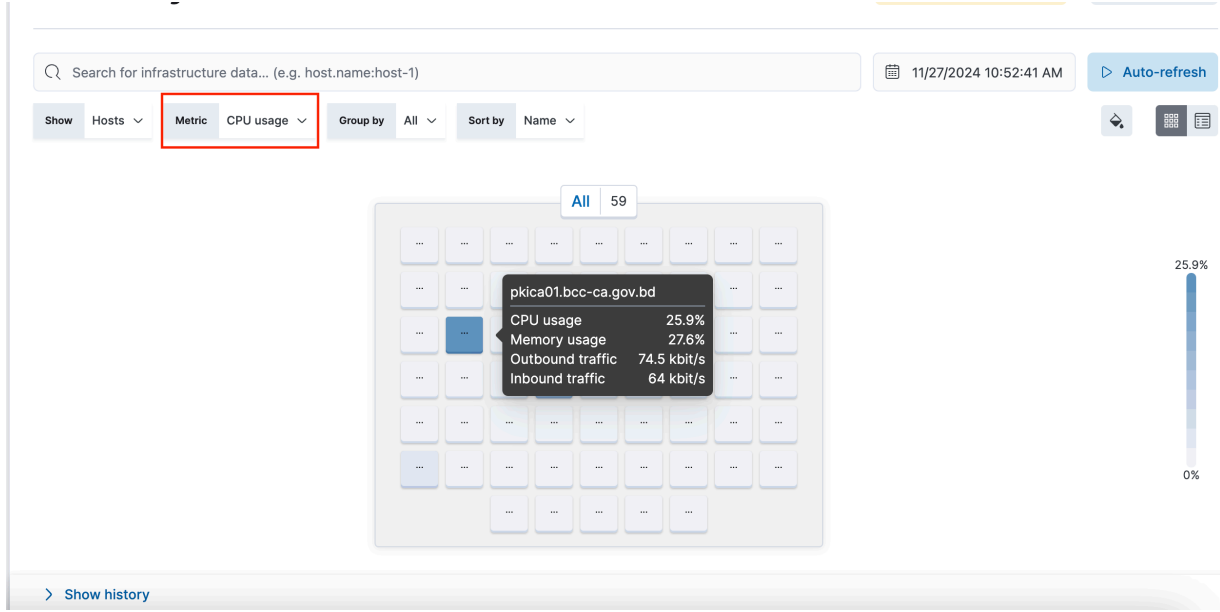
BANGLADESH COMPUTER COUNCIL

What to monitor

- Metrics
  - CPU
  - Memory
  - Disk
  - Network
- Logs
- Traces

BANGLADESH COMPUTER COUNCIL

Inventory


Search for infrastructure data... (e.g. host.name:host-1)
11/27/2024 10:52:41 AM
Auto-refresh
Show
Hosts
Metric
CPU usage
Group by
All
Sort by
Name
All
59
pkica01.bcc-ca.gov.bd
CPU usage 25.9%
Memory usage 27.6%
Outbound traffic 74.5 kbit/s
Inbound traffic 64 kbit/s
25.9%
0%
Show history

Log Ingestion (Demo)

Welcome to Elastic

Username

Password

Log in

BANGLADESH COMPUTER COUNCIL

Welcome to Elastic

Username

Password

Log in

BANGLADESH COMPUTER COUNCIL

Custom Dashboard for VMs

BANGLADESH COMPUTER COUNCIL

Single VMs

Welcome to Elastic

Username

Password

Log in

BANGLADESH COMPUTER COUNCIL

Multiple VMs

Welcome to Elastic

Username

Password

Log in

BANGLADESH COMPUTER COUNCIL

Applications Performance Monitoring

BANGLADESH COMPUTER COUNCIL

APM Tools

- Counters
- Profiling
- Tracing

BANGLADESH COMPUTER COUNCIL

1. Counters

- Real-time **metrics** tracking for **key performance indicators**
- Monitors system vitals like **requests/sec, error rates, and resource usage**
- Enables **threshold-based alerting** and trend analysis

2. Profiling

- Deep dive analysis of application runtime behavior
- **CPU, memory, and thread usage examination**
- Identifies performance bottlenecks and resource-intensive operations

3. Tracing

- End-to-end visibility of request flow through distributed systems
- Tracks latency across **service boundaries and components**
- Helps diagnose issues in microservices architectures

Counters

These are like the gauges in your car's dashboard. They constantly measure and record specific metrics in your application:

- Request rates: How many users/calls your system handles per second
- Error rates: How often things go wrong (like 404s or 500s)
- Response times: How fast your system responds
- Resource usage: CPU, memory, disk space consumption
- Custom business metrics: Like number of orders processed

Counters are great for real-time monitoring and alerting. For example, if your error rate suddenly spikes above 1%, you can get an immediate notification.

Profiling

Think of profiling as taking an X-ray of your application while it's running. It reveals:

- Which functions are taking the most time to execute
- Memory allocation and garbage collection patterns
- CPU usage across different threads
- Database query performance
- Hot spots in your code that need optimization

For example, profiling might reveal that a particular database query is taking 80% of your response time, helping you identify exactly what to optimize.

Tracing

Imagine following a letter through the postal system - that’s what tracing does for requests in your system:

- Follows a single request as it moves through different services
- Records timing at each step
- Shows the relationships between services
- Helps understand the full journey of a request

For example, when a user experiences slow checkout, tracing can show that it’s because the payment service is slow, not the shopping cart service.

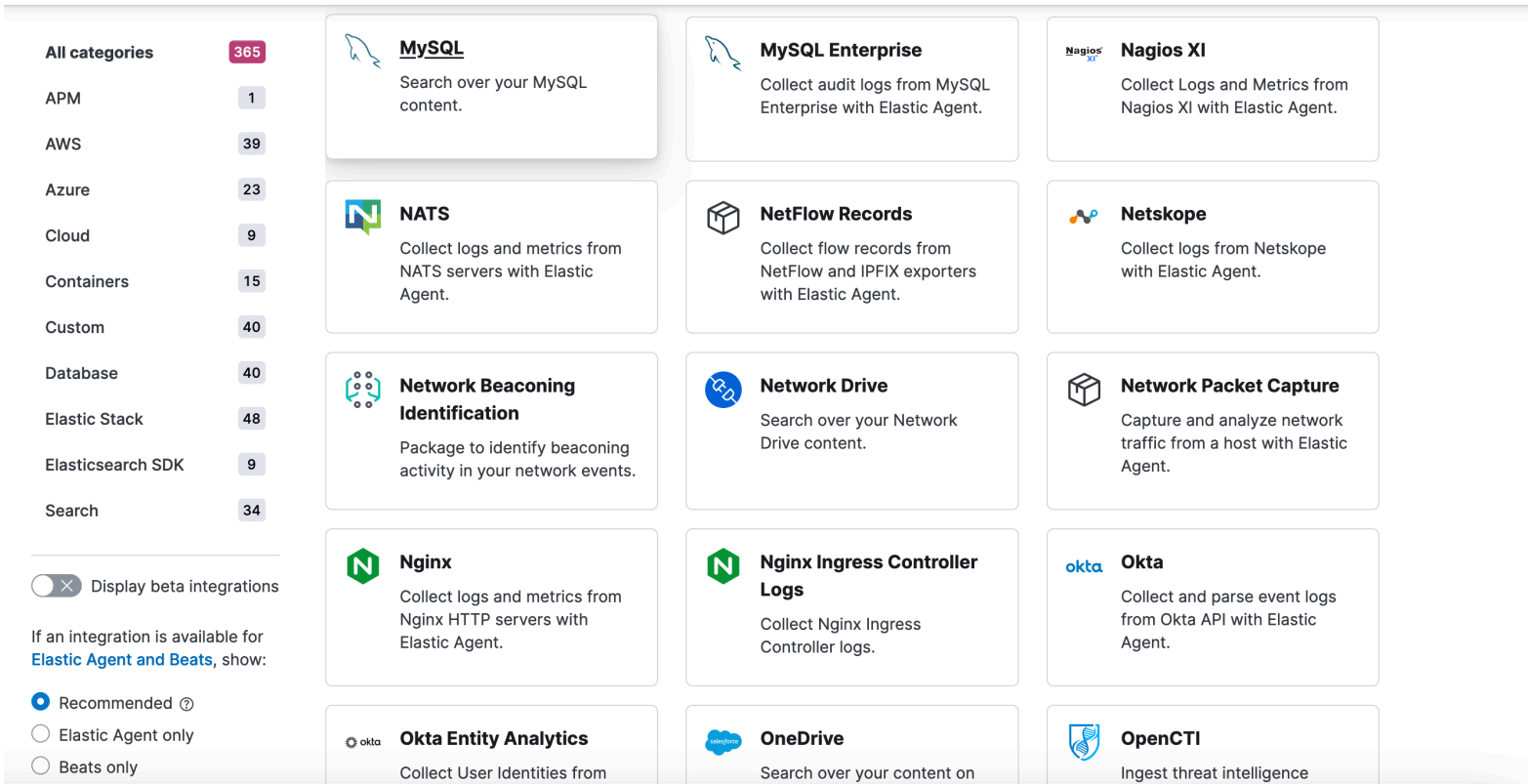
These three tools work together to give you complete visibility:

- Counters tell you THAT something is wrong
- Profiling tells you WHY something is wrong within a service
- Tracing tells you WHERE something is wrong across services

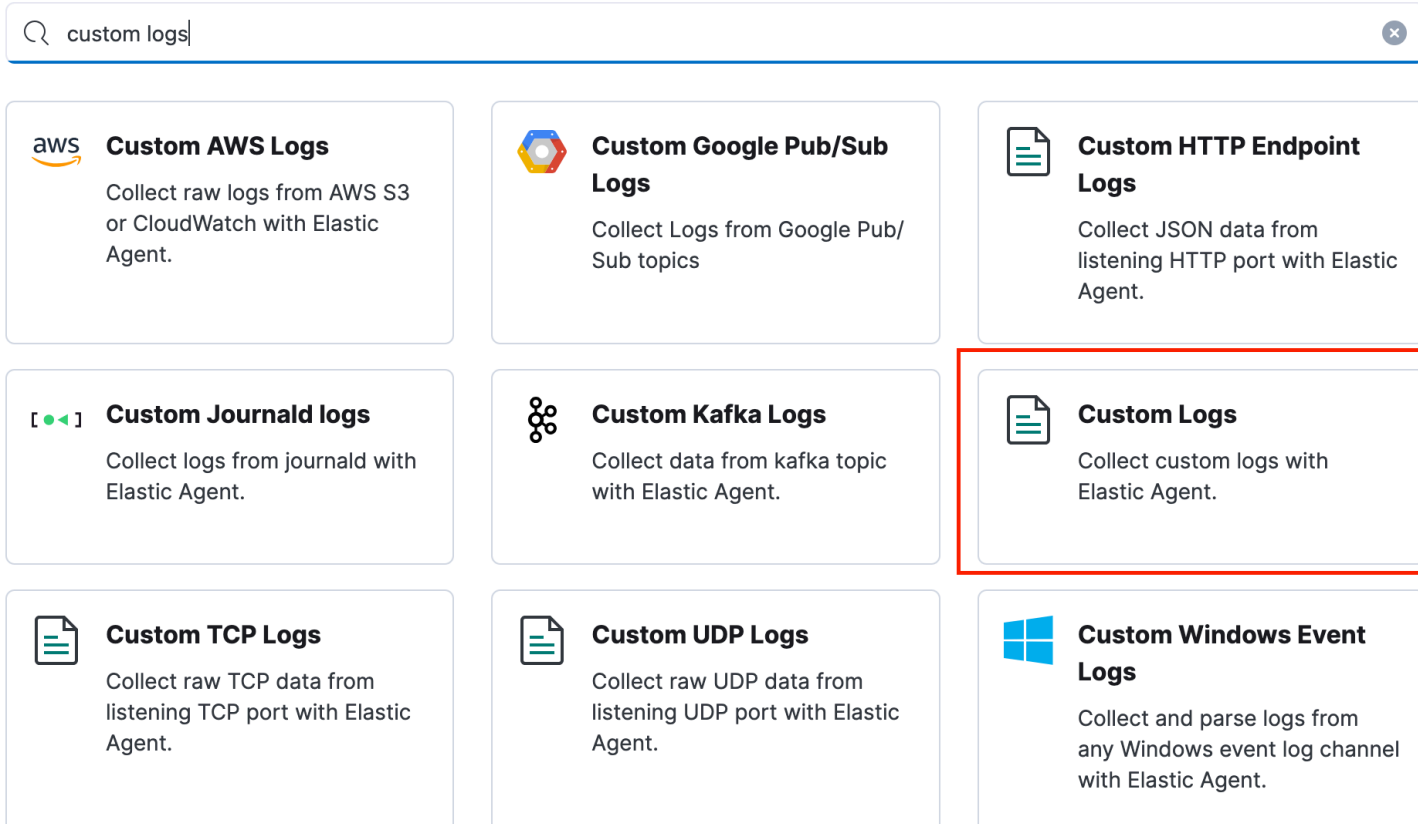
When combined, these tools make troubleshooting much faster and more effective. Instead of guessing what might be wrong, you have concrete data pointing to the exact issue.

Log Ingestion

Built in integrations


All categories 365
APM 1
AWS 39
Azure 23
Cloud 9
Containers 15
Custom 40
Database 40
Elastic Stack 48
Elasticsearch SDK 9
Search 34
Display beta integrations
If an integration is available for Elastic Agent and Beats, show:
Recommended
Elastic Agent only
Beats only
MySQL
Search over your MySQL content.
MySQL Enterprise
Collect audit logs from MySQL Enterprise with Elastic Agent.
Nagios XI
Collect Logs and Metrics from Nagios XI with Elastic Agent.
NATS
Collect logs and metrics from NATS servers with Elastic Agent.
NetFlow Records
Collect flow records from NetFlow and IPFIX exporters with Elastic Agent.
Netskope
Collect logs from Netskope with Elastic Agent.
Network Beaconing Identification
Package to identify beaconing activity in your network events.
Network Drive
Search over your Network Drive content.
Network Packet Capture
Capture and analyze network traffic from a host with Elastic Agent.
Nginx
Collect logs and metrics from Nginx HTTP servers with Elastic Agent.
Nginx Ingress Controller Logs
Collect Nginx Ingress Controller logs.
Okta
Collect and parse event logs from Okta API with Elastic Agent.
Okta Entity Analytics
Collect User Identities from
OneDrive
Search over your content on
OpenCTI
Ingest threat intelligence

Custom Logs


custom logs
Custom AWS Logs
Collect raw logs from AWS S3 or CloudWatch with Elastic Agent.
Custom Google Pub/Sub Logs
Collect Logs from Google Pub/Sub topics
Custom HTTP Endpoint Logs
Collect JSON data from listening HTTP port with Elastic Agent.
Custom Journald logs
Collect logs from journald with Elastic Agent.
Custom Kafka Logs
Collect data from kafka topic with Elastic Agent.
Custom Logs
Collect custom logs with Elastic Agent.
Custom TCP Logs
Collect raw TCP data from listening TCP port with Elastic Agent.
Custom UDP Logs
Collect raw UDP data from listening UDP port with Elastic Agent.
Custom Windows Event Logs
Collect and parse logs from any Windows event log channel with Elastic Agent.

Suppose we have a web server that is producing logs like this:

```
2024-02-05T09:47:36,919+06:00 INFO  com.ca.ekyc.activity.LogServiceImpl - Log Event - EventLog [id=0, entityId=-1, modul
2024-02-05T09:47:47,040+06:00 INFO  com.ca.ekyc.user.service.AdssUserService - ADSS GetUser Found - UserId:  0171529752
2024-02-05T09:47:47,138+06:00 INFO  com.ca.ekyc.activity.LogServiceImpl - Log Event - EventLog [id=0, entityId=-1, modul
2024-02-05T09:47:47,147+06:00 INFO  com.ca.ekyc.activity.LogServiceImpl - Log Event - EventLog [id=0, entityId=-1, modul
2024-02-05T09:47:47,154+06:00 INFO  com.ca.ekyc.interceptor.CommonApiInterceptor - CommonApiInterceptor  Completed : Use
2024-02-05T09:47:47,457+06:00 INFO  com.ca.ekyc.api.AuthController - Login Request : Username = bcc-esign-api-user
2024-02-05T09:47:47,461+06:00 INFO  c.ca.ekyc.security.services.UserDetailsServiceImpl - Username: bcc-esign-api-user
2024-02-05T09:47:47,553+06:00 INFO  com.ca.ekyc.activity.LogServiceImpl - Log Event - EventLog [id=0, entityId=2, module
2024-02-05T09:47:47,561+06:00 INFO  com.ca.ekyc.interceptor.CommonApiInterceptor - CommonApiInterceptor  Completed : Use
2024-02-05T09:47:47,571+06:00 INFO  com.ca.ekyc.security.jwt.AuthTokenFilter - username: bcc-esign-api-user roles: [ROLE
2024-02-05T15:08:35,622+06:00 ERROR com.ca.ekyc.api.RestSigningController - Get Certificate Error - UserId doesn't exist
java.lang.Exception: UserId doesn't exist: 01517109340
        at com.ca.ekyc.user.service.AdssCertificateService.getUserCertificate(AdssCertificateService.java:188)
        at com.ca.ekyc.api.RestSigningController.getUserCertificate(RestSigningController.java:96)
        at com.ca.ekyc.api.RestSigningController$$FastClassBySpringCGLIB$$2ca0c3e.invoke(<generated>)
        at org.springframework.cglib.proxy.MethodProxy.invoke(MethodProxy.java:218)
        ...
        ...
```

Ingest Pipeline

Multiline Configration

```
multiline.pattern: '^[0-9]{4}-[0-9]{2}-[0-9]{2}\s[0
multiline.negate: true
multiline.match: after
multiline.timeout: 15s
multiline.max_lines: 5000
```

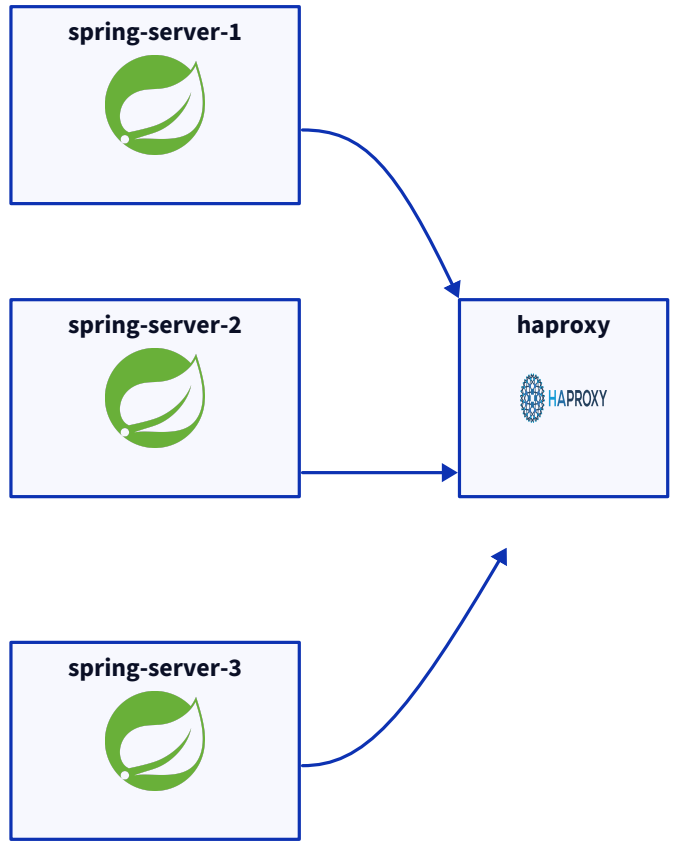
BANGLADESH COMPUTER COUNCIL

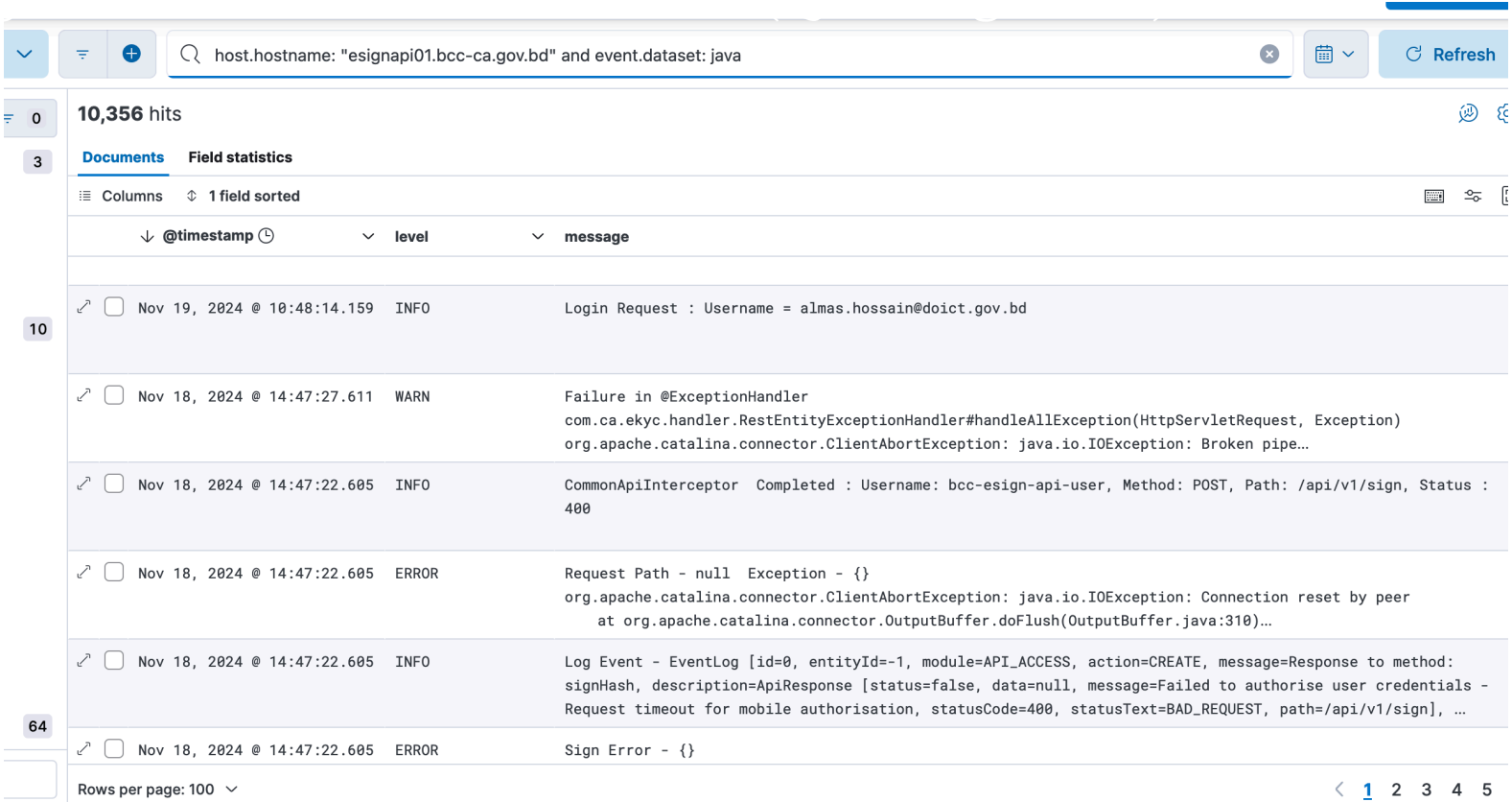
Processor

```
[
  {
    "grok": {
      "field": "message",
      "patterns": [
        "%{DATA:timestamp} \\[%{DATA:thread}\\] %{LOGLEVEL:level} %{JAVAFILE:class} - %{GREEDYMULTILINE:message}"
      ],
      "pattern_definitions": {
        "GREEDYMULTILINE": "(.|\n)*"
      }
    }
  }
]
```

Binding Processor to Custom Logs Integration

```
pipeline: logs-signinghub-pipeline
```


spring-server-1
spring-server-2
spring-server-3
haproxy
HAPROXY


host.hostname: "esignapi01.bcc-ca.gov.bd" and event.dataset: java
Refresh
10,356 hits
Documents
Field statistics
Columns
1 field sorted
@timestamp
level
message
Nov 19, 2024 @ 10:48:14.159 INFO Login Request : Username = almas.hossain@doict.gov.bd
Nov 18, 2024 @ 14:47:27.611 WARN Failure in @ExceptionHandler com.ca.ekyc.handler.RestEntityExceptionHandler#handleAllException(HttpServletRequest, Exception) org.apache.catalina.connector.ClientAbortException: java.io.IOException: Broken pipe…
Nov 18, 2024 @ 14:47:22.605 INFO CommonApiInterceptor Completed : Username: bcc-esign-api-user, Method: POST, Path: /api/v1/sign, Status : 400
Nov 18, 2024 @ 14:47:22.605 ERROR Request Path - null Exception - {} org.apache.catalina.connector.ClientAbortException: java.io.IOException: Connection reset by peer at org.apache.catalina.connector.OutputBuffer.doFlush(OutputBuffer.java:310)…
Nov 18, 2024 @ 14:47:22.605 INFO Log Event - EventLog [id=0, entityId=-1, module=API_ACCESS, action=CREATE, message=Response to method: signHash, description=ApiResponse [status=false, data=null, message=Failed to authorise user credentials - Request timeout for mobile authorisation, statusCode=400, statusText=BAD_REQUEST, path=/api/v1/sign], …
Nov 18, 2024 @ 14:47:22.605 ERROR Sign Error - {}
Rows per page: 100
1 2 3 4 5

Welcome to Elastic

Username

Password

Log in

BANGLADESH COMPUTER COUNCIL

Sample Dashboard (QuickSign-API)

Welcome to Elastic

Username

Password

Log in

BANGLADESH COMPUTER COUNCIL

Sample Dashboard (BCC-CA-Boot)

Welcome to Elastic

Username

Password

Log in

BANGLADESH COMPUTER COUNCIL

Synthetics

Welcome to Elastic

Username

Password

Log in

BANGLADESH COMPUTER COUNCIL

Alerts

Learn More

Documentation · GitHub · Showcases

Powered by Slidev