

<https://stackshare.io/elk>

Splunk Architecture:

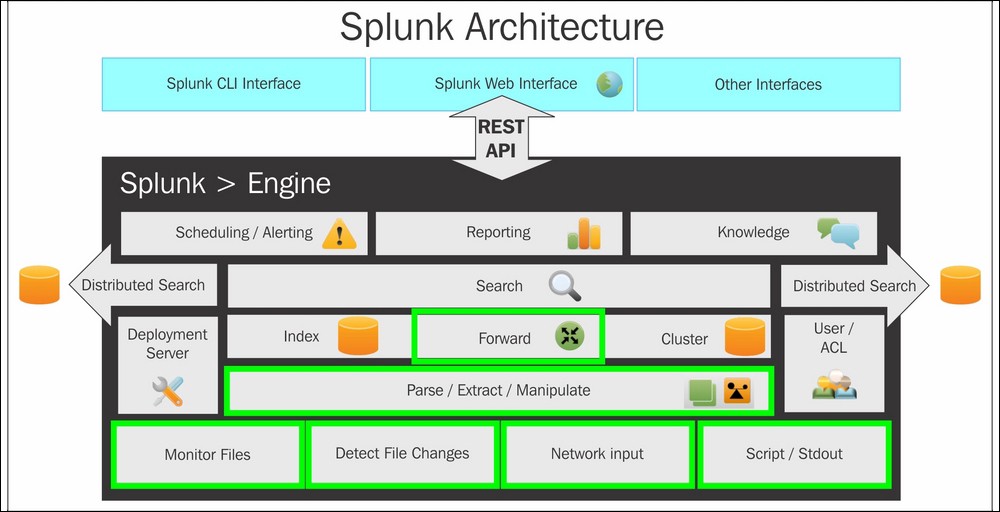
Splunk was introduced in 2003 and is a paid tool. It helps in analyzing structures as well as semi-structured data. Splunk is also known as the “Google for log files.” Splunk is a tool that processes every type of log being generated within an organization. It is a software platform that search, visualize, monitor, and analyze the data generated by the machine in real-time. Splunk refines the data to create powerful insights into your log data with charts, alerts, graphs, etc.

It is one of the popular DevOps tools used in the market. Its benefits are not only limited to the log management and analysis solution but also ensures security and management events. Splunk performs security analysis and assessment of the gathered logs to get metrics about the organization’s performance. Splunk uses SPL (search processing language) to serve and execute queries on large and complex data sets.

Splunk comes with three major components- Forwarder, Indexer, and Search head. Forwarder helps in pushing the data to the remote indexer. Then indexer responds to the search queries. Search head is a front-end web user interface where all the three components are combined for better visibility.

***Splunk has the below features.***

* It helps in accelerating the development and testing of applications.
* It helps you to create real-time data applications.
* It helps you to generate ROI faster.
* It comes with search, optimization, and visualization capability for every type of user.



Pros and cons:

<https://www.trustradius.com/products/splunk-enterprise/reviews?qs=pros-and-cons>

**Goods:**

* Very easy to Use
* Good customer support
* Supports collection of a lot of types of data.
* Provides good security
* Has good customizability based on user requirements
* ML abilities
* Log mining

**Bads:**

* It's very costly
* Steep learning curve
* Complex architecture

Best Sue Cases:

<https://igorgarofano.wordpress.com/2018/04/03/splunk-top-five-use-case/>

<https://www.guru99.com/elk-stack-tutorial.html>

ELK Pros and Cons:

Pros:

Free to Get Started

One of the key reasons for the growth in popularity of the ELK stack is its low financial barrier to entry. All of the software components of ELK are free and open-source - that means no up-front purchases are required and there are no ongoing software licensing fees.

Multiple Hosting Options

When it comes to deploying an ELK stack, organizations have multiple hosting options to choose from. For organizations with the right capabilities and resources, an ELK stack can be installed on a local server and managed in-house. Alternatively, organizations can choose to deploy their ELK stack as a managed service with products like Amazon Elasticsearch Service by partnering with a specialist MSP.

Centralized Logging Capabilities

One of the most important features of the ELK stack is that it offers centralized logging capabilities, allowing users to aggregate logs from increasingly complex cloud environments into a single searchable index. This capability makes it possible to correlate log and event data from multiple sources, enabling use cases like security monitoring and root cause analysis.

Real-Time Data Analysis & Visualization

With Kibana, ELK stack users can create data visualizations and build custom dashboards using real-time data from Elasticsearch. The ability to visualize data in real time decreases time-to-insights, supporting a variety of use cases and driving organizational agility and informed decision-making.

Official Clients in Multiple Programming Languages

Some ELK stack users have multiple languages in their codebase and wish to use Elasticsearch from all of them. To support this requirement, developers at Elastic have released official clients for Elasticsearch in at least 12 programming languages, including JavaScript, Go, Python, .NET, and Perl. Elastic provides support for all of its official clients, fixing bugs and responding to support queries as needed.

Advantages of ElasticSearch:

1. Elasticsearch is developed on Java, which makes it compatible on almost every platform.
2. Elasticsearch is real-time, in other words after one second the added document is searchable in this engine.
3. Elasticsearch is distributed, which makes it easy to scale and integrate in any big organization.
4. Elasticsearch is creating full backups in an easy way with the concept of gateway, which is present in Elasticsearch.
5. Handling multi-tenancy is very easy in Elasticsearch when compared to Apache Solr.
6. Elasticsearch uses JSON objects as responses, which makes it possible to invoke the Elasticsearch server with a large number of different programming languages.
7. Elasticsearch supports almost every document type except those that do not support text rendering.

Elasticsearch – Disadvantages

1. Elasticsearch does not have multi-language support in terms of handling request and response data (only possible in JSON) unlike in Apache Solr, where it is possible in CSV, XML and JSON formats.
2. Elasticsearch also has a problem with Split-brain situations but in rare cases.

Use cases:

### #1 – Logging and Log Analysis

For anyone familiar with Elasticsearch, this one should be no surprise. The ecosystem built up around Elasticsearch has made it one of the easiest to implement and scale logging solutions. Many of the the users on our platform are no different and have taken advantage of this to either add logging to their main use case, or are using us purely for logging. From [Beats](https://www.elastic.co/products/beats), to [Logstash](https://www.elastic.co/products/logstash), to [Ingest Nodes](https://www.elastic.co/guide/en/elasticsearch/reference/current/ingest.html), Elasticsearch gives you plenty of options for grabbing data wherever it lives and getting it indexed. From there, tools like [Kibana](https://www.elastic.co/products/kibana)give you the ability to create rich dashboards and analysis, while [Curator](http://objectrocket.com/blog/elasticsearch/elasticsearch-curator)allows you to put the retention period on autopilot.

### #2 – Scraping and Combining Public Data

Like log data, the Elastic Stack has plenty of tools to make grabbing and indexing remote data easy. Also, like most document stores, the lack of a strict schema gives Elasticsearch the flexibility to take in multiple different sources of data and still keep it all manageable and searchable. A cool example of this that you can check out is our [Twitter connector](http://objectrocket.com/docs/twitter_to_elasticsearch.html), which allows you to set up hashtags to watch on Twitter and then grab all tweets with those hashtags and analyze them in Kibana. We built that product on core Elastic Stack components and added some additional pieces to help it scale.

### #3 – Full Text Search

It’s also no surprise that full text search, as the core capability of Elasticsearch, is high on this list. The surprising part is the applications of this among our customer set, which go well beyond traditional Enterprise search or E-commerce. From fraud detection/security to collaboration and beyond, our users have shown that Elasticsearch’s search capabilities are powerful, flexible, and include a great number of tools to make search easier; Elasticsearch has its own query DSL as well as built in capabilities for auto-complete, “Did you mean” responses, and more.

### #4 – Event Data and Metrics

Elasticsearch also operates really well on time-series data like metrics and application events. This is another area where the huge Beats ecosystem allows you to easily grab data for common applications. Whatever technologies you use, there’s a pretty good chance that Elasticsearch has the components to grab metrics and events out of the box… and in the rare case that it can’t, adding that capability is really easy.

### #5 – Visualizing Data

With tons of charting options, a tile service for geo-data, and TimeLion for time-series data, Kibana is an amazingly powerful and easy to use visualization tool. For every use case above there is some visual component handled by Kibana. Once you’re comfortable with the various data ingest tools, you’ll find that Elasticsearch + Kibana will become your go-to tool for visualizing data that you’re trying to wrap your head around.

<https://www.objectrocket.com/blog/elasticsearch/top-elasticsearch-use-cases/>

<https://dzone.com/articles/elk-stack-overview-and-the-need-for-it>

<https://www.elastic.co/customers/success-stories?usecase=enterprise-search>

<https://blog.scottlogic.com/2016/05/26/big-data-logging-elk-stack.html>

<https://volansys.com/case-study/network-monitoring-solution-using-elk-stack/>

App Dynamics:

<https://docs.appdynamics.com/display/PRO21/AppDynamics+Application+Performance+Monitoring+Platform>

<https://docs.appdynamics.com/display/PRO14S/Architecture>

Best Use cases:

<https://www.appdynamics.com/aiops/aiops-use-cases#~current-aiops-use-cases>

<https://docs.appdynamics.com/display/PRO21/Workflows+and+Example+Use+Cases>

<https://docs.appdynamics.com/display/PRO21/Example+Use+Cases>

<https://www.appdynamics.com/customers>

<https://www.appdynamics.com/case-study/cisco>

Slueth and zipkin:

* **Spring Cloud Sleuth** is used to generate and attach the trace id, span id to the logs so that these can then be used by tools like Zipkin and ELK for storage and analysis
* **Zipkin** is a distributed tracing system. It helps gather timing data needed to troubleshoot latency problems in service architectures. Features include both the collection and lookup of this data.

