Dieser Artikel beschreibt exemplarisch das Logmanagement unter Verwendung des ELK-Stacks. Zum Thema Logmanagement siehe auch den Artikel: „Wozu ein zentrales Logmanagement?“

Log-Strategie

Zunächst ist die Logfilestrategie festzulegen, d.h. zu klären, welche Logfiles zu betrachten sind und wie diese voneinander abhängen. Dazu sind folgende Fragen zu stellen:

- Welche Logfiles auf welchen Rechnern sind relevant oder könnten für die Auswertungen relevant werden?
- Welche Daten aus den Logfiles könnten mich interessieren?
- Welche von diesen Daten korrelieren miteinander?

Log-Architektur

Abbildung 1 zeigt einen möglichen Aufbau der ELK-Log-Architektur (Production ready).

Die horizontale Skalierung ist auf jeden Fall ab (8) sichergestellt. Skalierung für (4) und (6) ist grundsätzlich möglich, aber aufwändiger, da im Log Courier unterschiedliche Ziele bzw. mehrere Logstash-Indexer, die auf verschiedene Redis-Instanzen gehen. Redis braucht kaum Speicher, den ganzen Hauptspeicher benötigt Elastic Search. Grundsätzlich mehrere Worker (mehrere Threads oder IO-basiert).

Produktives Beispiel: Der Server hat 64 GiB RAM, davon sind 32 GiB belegt. ElasticSearch beansprucht ca 10 GiB RAM und 85 GiB Disk Space, der bei steigender Requestzahl entsprechend ansteigt. Derzeit fallen täglich ca. 3 Millionen Logeinträge an. Aufgrund der Retention Policy werden immer genau 30 Tage vorgehalten, wodurch die Speichermenge begrenzt wird.

Log Courier

Log Courier ist grundsätzlich ein Agent zur Übertragung der Logdaten an den Logstash-Shipper.

Log Courier wird pro Maschine, für die Logs auszuwerten sind, eingerichtet. Es handelt sich dabei um einen kleinen, handlichen Agenten, der in Go geschrieben ist und z.B. Multiline-Unterstützung bietet (z.B. Stack-Traces). Log Courier ist leichtgewichtiger und performanter als Logstash und wird hier deshalb bevorzugt. Von elastic.co gibt es mittlerweile auch Beats als Alternative zu Log Courier – auch leichtgewichtig und Multiline-fähig. Mittlerweile auch Logstash Multiline-fähig, aber bzgl. des Footprints nicht leichtgewichtig.

Der Logstash-Shipper

Log Courier überträgt die Logdaten an eine Logstash-Instanz, die als Shipper verwendet wird. Um die Daten entgegennehmen zu können, verwendet Logstash ein Log-Courier-Plugin, mit dem Logstash die Daten über das binäre Log-Courier-Protokoll entgegennehmen kann.

Der Shipper, der mit Logstash realisiert wird, ist dazu da, die Logdaten, die über das Log-Courier-Protokoll ankommen, an Redis weiterzugeben. Der Logstash-Shipper hat dabei eigentlich keine Logik außer „Drop Empty“, d.h. leere Logeinträge werden weggeworfen. Sollte eigentlich auch Log Courier können, wenn man aber Multiline und Codec gleichzeitig verwendet, gibt es mit Log Courier ein Problem.

Der Logstash-Shipper macht nichts anderes, als die Logdaten an Redis weiterzugeben.

- Für Input: Courier-Plugin
- Filter: z.B. leere Log-Events verwerfen
- Output: Redis-Plugin

Redis

Redis wird als In-Memory-Zwischenspeicher und Skalierungsmittel verwendet. Redis verwaltet die Logdaten also in-memory in einer Liste und stellt sie so für die Verarbeitung durch den Logstash-Indexer bereit.

Der Logstash-Indexer

Der Logstash-Indexer ist eine weitere Logstash-Instanz, die zur Aufbereitung der Log-Events verwendet wird. Dazu holt er sich einzelne Sätze im Pull-Verfahren aus Redis und zerlegt diese in einzelne Felder unter Verwendung der Tags. In der Datei logstash-indexer.conf ist beschrieben, wie ein Satztyp in einzelne Felder zerlegt wird. Dann schiebt der Logstash-Indexer die aufbereiteten Daten in das ElasticSearch.

ElasticSearch

Über ElasticSearch erfolgt die Datenhaltung, Indizierung und Suche. Die Felder werden dazu in ElasticSearch indiziert und stehen dann für Kibana zur Suche zur Verfügung.

Kibana

Kibana ist ein graphisches Dashboard-Tool, über das Abfragen und Auswertungen abgesetzt werden können.

Beispielgrafik: Links Top-10-Error Messages. Mitte: Static vs. Dynamic Content bzgl. der Requests, rechts: Dynamic Content aufgesplittet.

Der Standardanwendungsfall ist das Anzeigen von Logdaten. Durch Einstellen des Zeitbereichs und das Setzen von Filtern wird das Ergebnis immer weiter eingegrenzt, bis man die gewünschte Menge an Logmeldungen vor sich hat.

Beim erweiterten Anwendungsfall baucht man für wiederkehrende Abfragen oder KPIs Dashboards, wobei hierzu mächtige Visualisierungen und Diagrammdarstellungen zur Anwendung kommen.

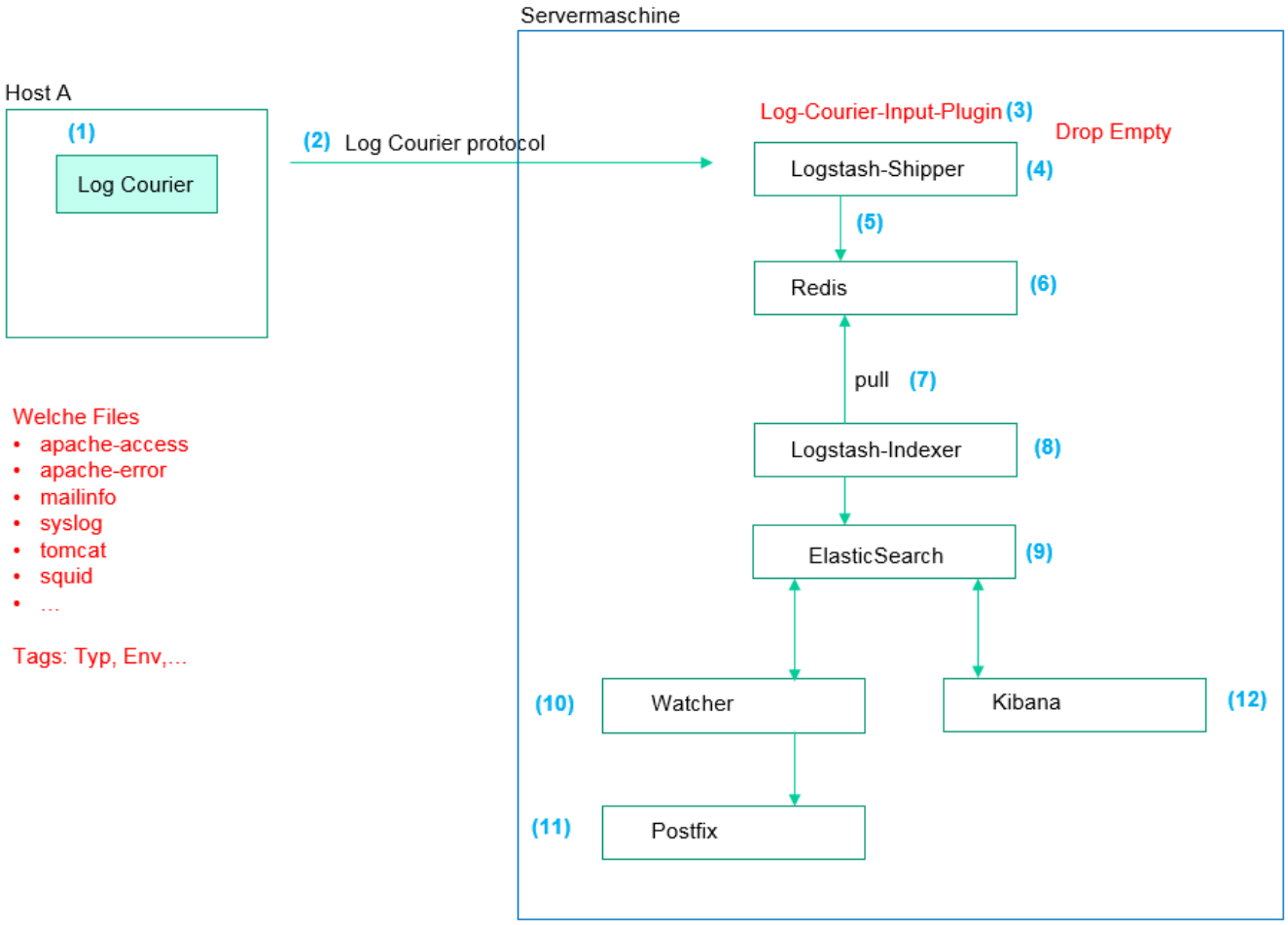

Abbildung 1: Die ELK-Log-Architektur

(1) Log Courier wird auf den Hosts, auf denen Logdaten anfallen, ausgeführt.

(2) Die Übertragung der Logdaten an den Logstash-Shipper erfolgt über das Log-Courier-Protokoll (binär).

(3) Logstash verwendet das Log-Courier-Input-Plugin.

(4) Der Logstash-Shipper hat eigentlich keine Logik außer „Drop Empty“, d.h. leere Logeinträge verwerfen. Sollte eigentlich auch Log Courier können, wenn man aber Multiline und Codec gleichzeitig verwendet, gibt es ein Problem.

(5) Der Logstash-Shipper gibt die Logdaten an Redis weiter.

(6) Redis verwaltet die Logdaten in einer Liste.

(7) Der Logstash-Indexer liest die Logdaten aus Redis (pull).

(8) Der Logstash-Indexer zerlegt die Logdaten in Fields. In seiner Konfiguration werden Patterns definiert, die die Logdaten zerlegen.

(9) Neben den Rohdaten indiziert ElasticSearch die Fields. Die indizierten Felder können bei Abfragen effizient verwendet werden.

(10,11) Optionales Alerting (kostenpflichtig)

(12) Frontend für die Auswertungen.