Log-Management mit Loki

Fabian Thorns

Adminstammtisch Berlin, 4. Dezember 2025

Hi, ich bin Fabian

- **Fabian Thorns**
  - Berater, Trainer und Mitgründer der xamira networks GmbH:
    - Linux, Ansible, IPv6, Hochverfügbarkeit
    - Docker, Kubernetes, Helm, Argo CD, Prometheus, Loki, ...
  - M.Sc. Wirtschaftsinformatik
  - LFCE, LPIC-3, PCA, RHCS Containers for Kubernetes
  - Kubestronaut (KCNS, KCSA, CKAD, CKA, CKS)
  - Autor "IPv6-Handbuch"

Zentrales Log-Management

-- MARK --

Zentrales Log-Management

- Klassisches Log-Management:
  - Ist doch ganz einfach:
    - `journalctl`
    - `less`
    - `grep`
  - Next Level Stuff:
    - `grep` ⇨ `wc` ⇨ Cron ⇨ Email
    - `awk` ⇨ CSV ⇨ Tabellenkalkulation
  - Skalierung:
    - Wie zuvor, per SSH

Zentrales Log-Management

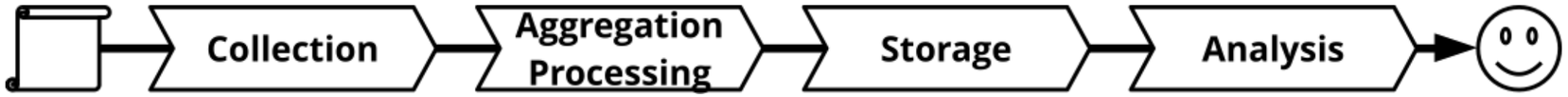
- Aber was ist mit …
  - … 500 Pods …
  - … auf 16 Kubernetes-Nodes …
  - … die mehrfach täglich neu erstellt werden?
- Nicht euer Thema?
- Okay, was ist mit …
  - … 40 virtuellen Maschinen …
  - … die jeweils mehrere Dienste beherbergen …
  - … die miteinander interagieren?
- Was, wenn das keine virtuellen Maschinen, sondern Router sind?

Zentrales Log-Management

- Vorteile zentralen Log-Managements:
  - Ein zentraler Ort um alle Logs zu analysieren:
    - Einheitliche Abfragesprache
    - Ohne `root`-Rechte auf den betroffenen Systemen
  - Analysen über mehrere Logquellen hinweg:
    - Interaktionen von verschiedenen Systemen und Anwendungen betrachten
    - Muster von Angriffen erkennen
    - Ereignisse korrelieren
  - Lagebild über die Gesamtsituation
  - Einheitliche Speicherung, Auswertung, Aufbewahrung und Entsorgung

Komponenten in Logging-Stacks

- Es gibt eine ganze Reihe von freien und proprietären Logging-Stacks:
  - Alle erfüllen prinzipiell dieselben Aufgaben
  - Je nach Architektur unterschiedliche Komponenten an verschiedenen Orten
  - Die folgenden Komponenten finden sich (mit etwas wenn und aber) in den meisten Lösungen wieder:
    - Log Collectoren
    - Log Aggregatoren und Processoren
    - Log Storages
    - Log Analyser



Komponenten in Logging-Stacks

- **Log Collectoren:**
  - Logmeldungen erheben:
    - Inhalte ("Was steht drinnen?")
    - Metadaten ("Wer hat's wann wo wohin geschrieben?")
  - Typische Quellen:
    - Dateien
    - Systemd Journal
    - Windows EventLog
    - Sockets (beispielsweise syslog-Protokoll)
    - Container oder Pods
    - ...

Komponenten in Logging-Stacks

- **Log Collectoren:**
  - Relevante Quellen identifizieren:
    - Dateisystem-Globbing
    - Statische Konfiguration
    - Docker-Socket / Kuberetes-API
    - …
  - Erhobene Meldungen an einen Logaggregator senden:
    - Authentifikation
    - Buffering und Log-Rotation
    - …
  - Log Collectoren werden in der Breite ausgerollt – überall da, wo Logmeldungen entstehen

Komponenten in Logging-Stacks

- **Log Aggregatoren und Processoren:**
  - Logmeldungen von den Collectoren empfangen
  - Verarbeiten der Logmeldungen:
    - Extrahieren besonders wichtiger Informationen für spätere Indexierung
    - Entfernen sensibler Informationen, beispielsweise Anonymisierung von IP-Adressen
    - Anreichern, beispielsweise Informationen zur Lokalität von IP-Adressen
    - Überführen in das Format des Storages, beispielsweise in JSON-Dokumente
  - Weiterleiten der Logmeldungen zum Log-Storage
  - Log Aggregatoren werden zentral bereitgestellt – gegebenenfalls mehrfach in topologischer Nähe zu den Logquellen

Komponenten in Logging-Stacks

- **Log Storages**
  - Aufbereitete Logmeldungen von den Aggregatoren entgegennehmen
  - Aufbereitete Logmeldungen in geeigneter Weise speichern und bereitstellen:
    - Zugriffsgeschwindigkeit und -latenz?
    - Indexe?
    - Volltextsuche?
  - Ausführen von Abfragen gegen die gespeicherten Logmeldungen:
    - Abfragesprache zum Filtern und Durchsuchen von Logmeldungen
    - Volltextsuche ist teuer, viele Indexe auch
  - Log Storage wird einmal zentral bereitgestellt – gegebenenfalls als Cluster oder in Shards

Komponenten in Logging-Stacks

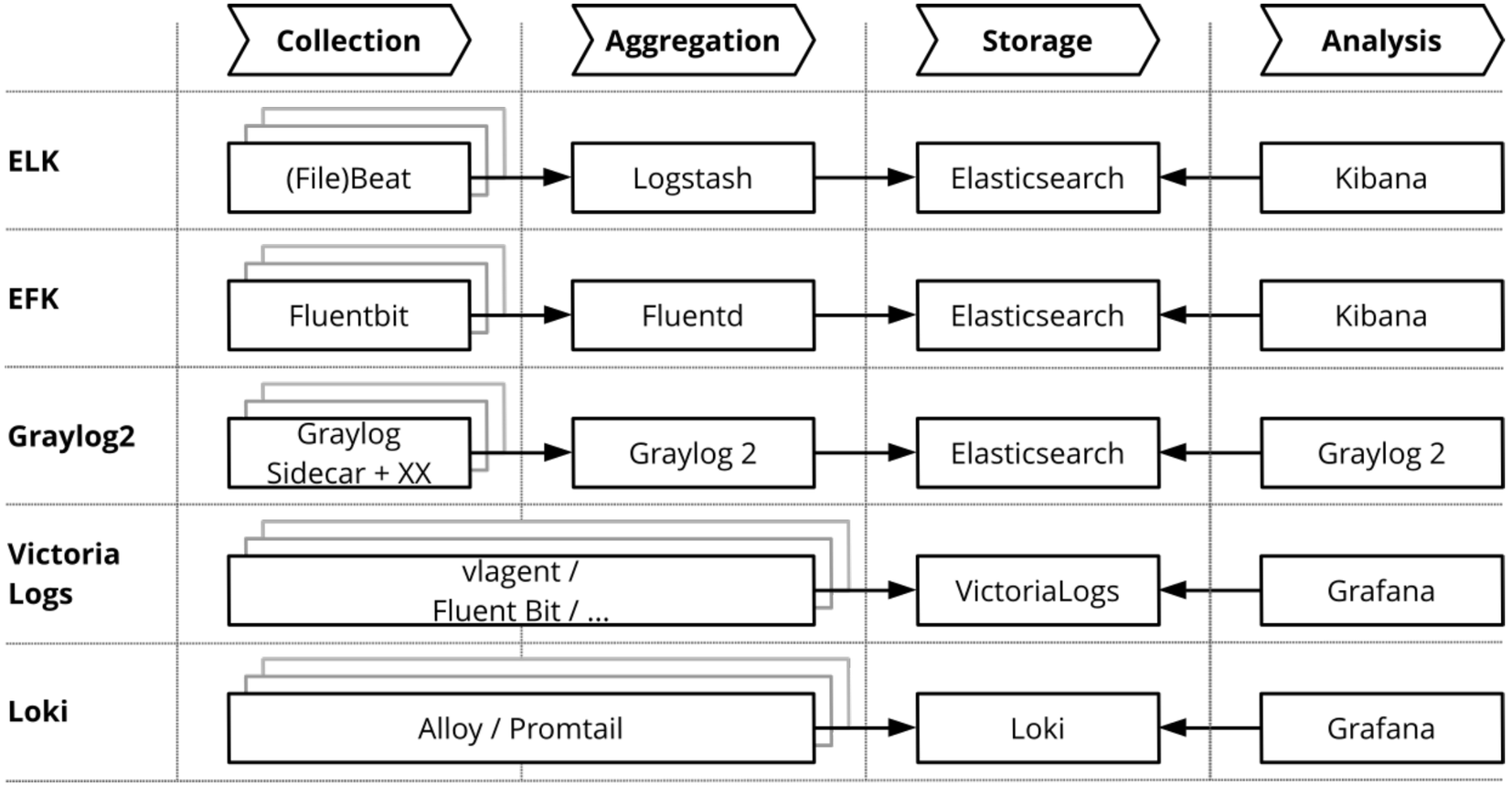
- **Log Analyser:**
  - Abfrage von Log-Inhalten aus dem Log Storage:
    - Betrachten aller Logs
    - Filtern und Durchsuchen der Meldungen
  - Visualisierung des Log-Aufkommens:
    - Grafische Darstellung des Volumens bestimmter Logs
    - Zielgerichtete Darstellung ausgewählter Inhalte
  - Verschiedene Use-Cases:
    - Explorative Ad-Hoc Analysen
    - Dashboards für Reporting und bekannte Fehlersituationen

Komponenten in Logging-Stacks

- **Log Analyser:**
  - Alarmierung:
    - Feststellen eines Alarmzustands:
      - Auftreten bestimmter (Mengen an) Meldungen
      - Ausbleiben bestimmter Meldungen
    - Gegebenenfalls Nutzung vorhandener Infrastruktur (beispielsweise Alertmanager)

Komponenten in Logging-Stacks

- Übersimplifiziert gibt es folgende gängige Open Source-Logging-Stacks:



- Vieles lässt sich auch anders kombinieren, ergänzen oder per Netzwerk antüdeln

Grafana Loki

Grafana Loki

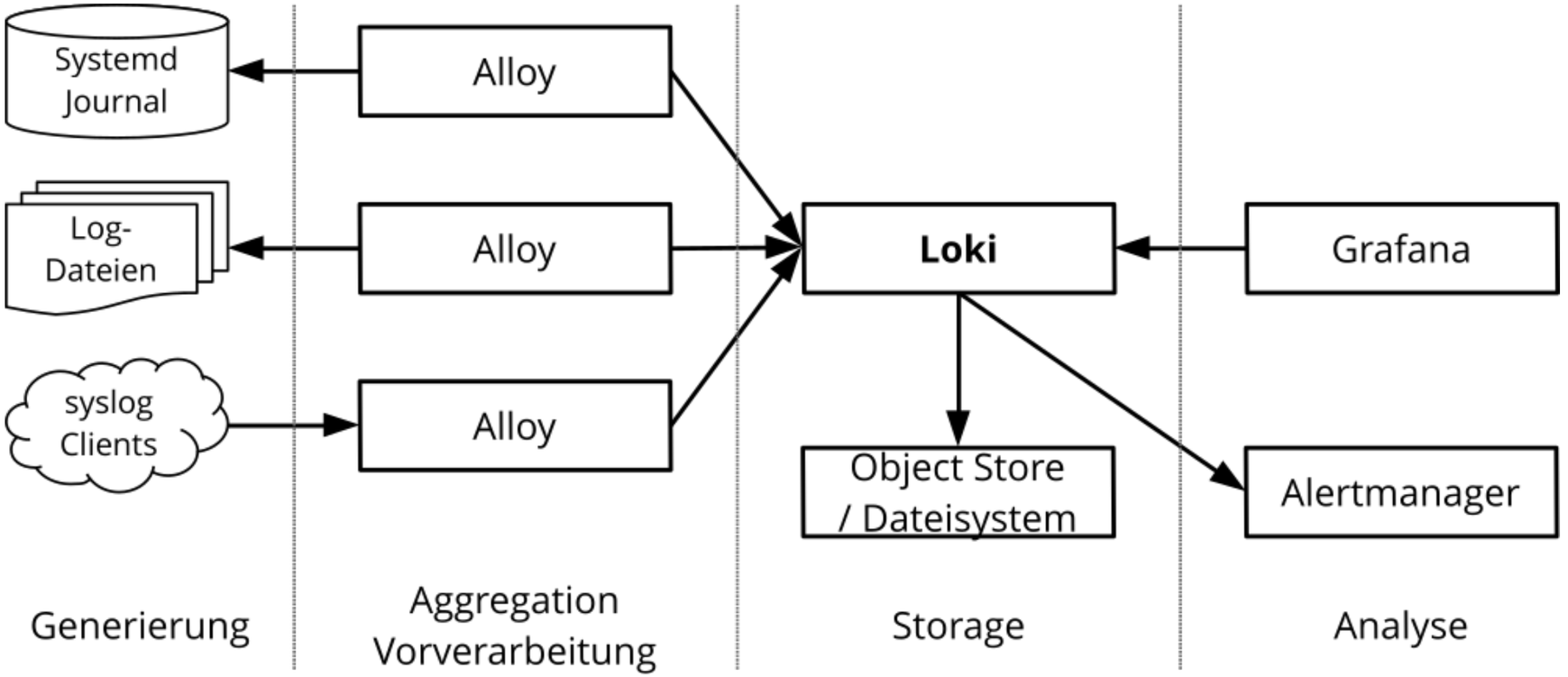
- **Loki ist ein Logmanagement-System:**
  - "Prometheus for Logs"
  - Speichert Logs im Dateisystem oder in S3-/GCS-kompatiblen Object Storage
  - Labels zur Indexierung und Suche, ähnlich wie Prometheus
  - Abfragesprache LogQL, ähnlich zu PromQL
- Grafana Labs in 2018
- Freie Software unter der AGPL version 3
- Angepasste Version in OpenShift
- Website: https://grafana.com/oss/loki/
- Dokumentation: https://grafana.com/docs/loki/latest/

Grafana Loki

- **Loki** selbst speichert Logs und führt Queries gegen den Datenbestand aus
- Für eine vollständige Log-Verarbeitung benötigt Loki weitere Komponenten:
  - **Alloy** (aktuell) oder **promtail** (veraltet) sammeln Logs ein, verarbeiten sie vor (parsen, umschreiben, anreichern, ...) und senden sie zu Loki
  - **Grafana** dient für den Zugriff auf Logs
  - **Object Storage** oder das **Dateisystem** speichert Logs
  - **Alertmanager** übermittelt Alarme
  - **Prometheus** kann die Metriken von Loki und Alloy überwachen

Grafana Loki

- Übersicht über die Logverarbeitung in Loki:



- Es gibt zahlreiche weitere Log-Quellen, insbesondere Container in Kubernetes-Pods
- Loki besteht aus internen Services, die auch getennt betrieben werden können

Grafana Loki

- Labels spielen eine zentrale Rolle in Loki:
  - Eine Reihe zusammengehörende Logmeldungen wird als **Stream** bezeichnet
  - Ein Stream durch eine eindeutige Kombination von **Labels** gekennzeichnet
  - Labels werden indexiert, so dass Logmeldungen nach Labels gefiltert werden können
  - **Anhand von Labels werden die Logmeldungen ausgewählt, die für eine konkrete Anfrage einer Volltextsuche unterzogen werden**
  - Fachlich sinnvolle Labels sind wichtig für die Performance von LogQL-Queries:
    - Zu wenige Labels führen zu unnötig umfangreichen Volltextsuchen
    - Zu hohe Kardinalität macht Loki langsam
  - Die Labels einer Logmeldung werden bei der Vorverarbeitung durch Alloy / promtail festgelegt und als Teil der Meldung an Loki übergeben

Ausprobieren!

Ausprobieren!

- Ich hab da mal was vorbereitet:
  - **Ubuntu-VM in einer Public Cloud**
  - **Global gültige IPv4- und IPv6-Adresse**
  - **SSH-Server auf Port 22 mit Passwort-Authentifikation und ohne fail2ban**
  - Grafana Paket-Repositories eingebunden
  - Alloy, Loki und Grafana installiert, abgesichert und grundkonfiguriert
  - Noch keinerlei Logverarbeitung eingerichtet
  - MaxMind GeoLite2-City Datenbank bereitgestellt

Demo 1 – Systemd-Journal in Loki

Ausprobieren!

- **Demo 1 – Systemd-Journal in Loki:**
  - Ziel:
    - Inhalte des Systemd-Journals sind über Loki zugänglich
    - Systemd Unit und Hostname werden als Label geführt
    - Zugriff über Grafana möglich

Demo 2 – SSH-Auswertung

Ausprobieren!

- **Demo 2 – SSH-Auswertung:**
  - Ziel:
    - Fehlerhafte SSH-Logins werden gesondert protokolliert
    - Die Meldungen haben ein eindeutiges Label, um sie zu finden
    - Das Land, aus dem IP-Adresse vermutlich stammt, wird als Label geführt
    - Im Inhalt der Meldung werden folgende Informationen so bereitgestellt, dass sie in Queries einfach verarbeitet werden können:
      - Die vollständige IP-Adresse
      - Die vermutliche Stadt der IP-Adresse
      - Der Username
    - Dashboard für die Auswertung nach häufigsten Usernamen und Quellländern

Das war's für heute!

Natürlich können Alloy, Loki und Grafana noch viel mehr...

Werbung

- Mehr davon? Gibt's bei uns!
- **Praxis-Trainings** für direkte Handlungsfähigkeit:
  - Docker, Kubernetes, Helm, Argo CD
  - Prometheus, **Loki**
  - IPv6, Linux-Grundlagen, Ansible
- **Missionsunterstützung und Consulting** zu all unseren Themen
- **www.xamira.de für weitere Informationen**
  - Offene Online-Kurse
  - Inhouse-Trainings mit individuellen Schwerpunkten
  - Wir können fast alles möglich machen – meldet euch!

Fragen?

Vielen Dank für eure Aufmerksamkeit!