# Vorstudie

## Anomalie Detection in Netzwerken

Ives Schneider

## Index

## 1. Management Summary

TODO

## 2. Projektinformationen

**Auftraggeber**

Technische Berufsschule Zürich
Sihlquai 101
8090 Zürich
admin.hf@tbz.zh.ch

**Projektleitung**

Ives Schneider
Binzstrasse 19
8712 Stäfa
ives.schneider@i-401.xyz

**Experte**

Marco Sieber
marco.sieber@tbz.ch

**Studiengang**

IT Service Engineer HF

**Klasse**

ITSE17a

## 2.1. Projektauftrag

### 2.1.1. Projektbeschreibung

Basierend auf dem Netzwerk der TBZ HF soll eine Lösung zur Anomalie Detection evaluiert werden und in einem POC aufgebaut und entsprechend dokumentiert werden (inkl. Betriebsdokumentationen).
Die Lösung soll die Fähigkeit bieten, unautorisierte Aktivitäten im Netzwerk erkennen zu können.

### 2.1.2. Problemstellung

Netzwerkanomalien können weitreichende folgen haben. Angefangen von kleineren Paketverlusten, bis hin zum Ausfall ganzer Komponenten.
Die Erkennung solcher Anomalien ist meist mühselig und schwierig und kann unter Umständen mehrere Ressourcen aufbrauchen bis der Standard wiederhergestellt werden kann.

### 2.1.3. Projektorganisation


*Figure 1. Projekt Mitarbeiter*

### 2.1.4. Projektzeitplan

TODO

### 2.1.5. Ausgangslage

TODO

# 3. Projektziele

Die Anforderungen sollen die nötigen Funktionen erfüllen, um eine gute Übersicht über das bestehende Netzwerk aufzuzeigen, Anomalien erkennen und deuten zu könenn.

## 3.1. Muss Kriterien

- Baseline muss umfänglich ersichtlich sein

  *Es muss eine grundlegende Analyse des normalen Netzwerkverkehrs erstellt werden.*
- Lösung soll Modular sein

  *Die Lösung soll mit mehreren NMS sowie Switches arbeiten können.*

  *Falls ein "Modul" nicht erwünscht sein sollte, kann es deaktiviert werden.*
- Scaleability soll vorhanden sein

  *Es soll die Möglichkeit bieten mehrere Log-Collectoren anzuschliessen.*
- OpenSource Lösung

  *Der Sourcecode soll für Erweiterungen veröffentlicht werden.*
- Betriebsdokumentation

  *Der Betrieb kann klar anhand einer Dokumentation nachvollzogen und nachgestellt werden.*
- Installationsdokumentation

  *Es ist eine Installationsdokumentation vorhanden, um Installationen ohne Experten nachzustellen.*
- Alerts müssen gemeldet werden

  *Anomalien werden via Mail an die zuständigen Administratoren gemeldet.*
- Web oder CLI Interface

  *Es ist ein Web und/oder CLI Interface vorhanden um die Applikation zu managen.*

## 3.2. Kann Kriterien

- Push Benachrichtigungen

  *Alerts können via Push notification abgesendet/empfangen werden.*
- SSO Authentifizierung

  *Single-Sign-On Anbindung an bestehende SSO Lösungen.*
- Multiarch

  *Die Lösung soll sowohl unter x86 sowie x86_64 laufen.*
- OS Independent

  *Die Lösung soll keine Abhängigkeit des unterliegenden Betriebssystem haben.*

## 3.3. Bewertungskriterien

- API

  *Die Applikation besitzt ein ausführliches und umfassendes API.*

- Dokumentation

  *API sowie andere Konfigurationsmöglichkeiten sind ausführlich dokumentiert.*

- Performance

  *Ein single-node muss auch bei höheren Lasten noch immer eine gute Performance liefern.*

- Skalierbarkeit

  *Clusterfunktionalität der Applikation.*

- Lizenz

  *Wie offen ist die Lizenz, kann die Applikation geändert werden?*

- Ausgereift

  Wie ausgereift ist die Applikation?

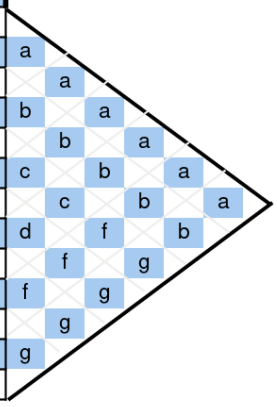
- Aktivität

  Wie aktiv wird an der Applikation weiterentwickelt?

## 3.4. Gewichtung

Die Gewichtung kann für beide evaluierten Software Lösungen eingesetzt werden.

| Gewicht | Rangfolge | Anzahl | | Kriterien |
|---|---|---|---|---|
| 28.571 | 1 | 6 | a | API |
| 23.81 | 2 | 5 | b | Dokumentation |
| 9.5238 | 5 | 2 | c | Performance |
| 4.7619 | 6 | 1 | d | Skalierbarkeit |
| 0 | 7 | 0 | e | Lizenz |
| 14.286 | 4 | 3 | f | Ausgereift |
| 19.048 | 3 | 4 | g | Aktivität |
| **Total** | **100** | | **21** | |

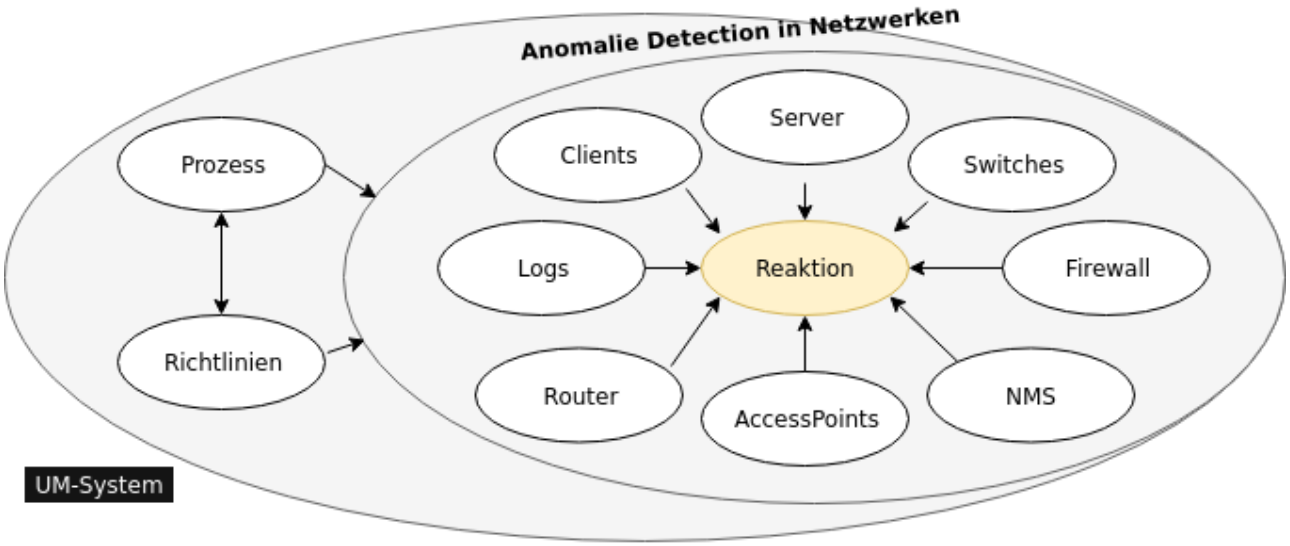


# 4. Analysetechnik

## 4.1. Abgrenzung

Um Anomalien frühzeitig und um möglichst "False Positives" zu vermeiden, ist es unabdingbar, möglichst viele Netzwerkgeräte miteinzubeziehen. Prozesse sowie bereits vorhandene Richtlinien werden durch das Projekt nicht abgeändert, sondern eventuell noch durch neue Prozesse ergänzt.

## 4.2. Systemgrenzen




### 4.2.1. Einflussgrössen / Restriktionen

Da das gesamte Netzwerk überwacht und Daten analysiert werden müssen, ist es wichtig, dass das Projekt transparent und mit den gegebenen Datenschutzrichtlinien im Einklang durchgeführt wird.
Grundsätzlich sollen keine bestehenden Arbeitsabläufe geändert oder gefährdet werden.

## 4.3. Analyse der Teilsysteme

### 4.3.1. Clients

Vorhandene Clients müssen kategorisiert und eingestuft werden.
Sobald ein Client hinzukommen oder entfernt werde sollte, muss eine Reaktion darauf ausgelöst werden und die Änderung in der Baseline vermerkt werden.

### 4.3.2. Server

Unbekannte Server und Services müssen gemeldet und eingestuft werden.
Ebenfalls sollten unbekannte Server direkt eine Reaktion auslösen.

### 4.3.3. Switches

Switches werden anhand von SNMP überwacht. Sollte sich ein Wert ausserhalb der Baseline befinden, muss eine Reaktion darauf erfolgen.

### 4.3.4. Firewall

Firewall werden mithilfe ihren APIs abgefragt. Falls kein API vorhanden sein sollte, wird auf SNMP zurückgegriffen.

### 4.3.5. NMS

Das NMS wird als einer der Hauptquellen für Informationen über das momentane Verhalten des Netzwerkes zu Rate gezogen.
Ebenfalls wird das NMS passiv sowie pro-aktiv in die Informationssuche miteingschlossen.

### 4.3.6. AccessPoints

Es soll stetts eine Übersicht über die Anzahl und Identifikationen der verbundenen Clients vorhanden sein.

### 4.3.7. Router

Router werden anhand ARP sowie SNMP in die Überwachung miteingebunden.

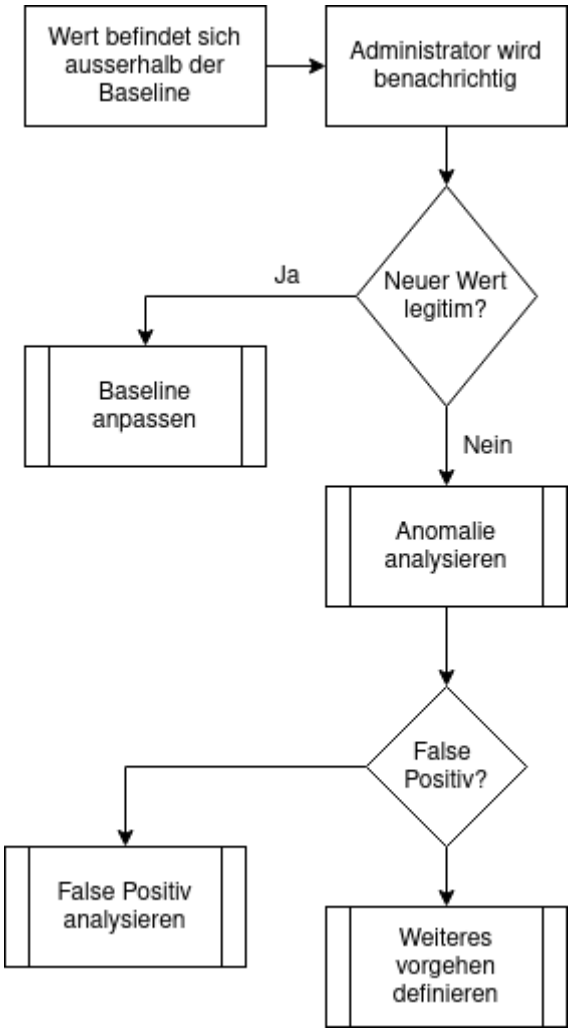
### 4.3.8. Logs

Es wird ein Log-Collector evaluiert um die gesamten Logs zentral zu speichern und abfragen zu können.

### 4.3.9. Reaktion

Da Abweichungen in der Baseline ein grosses Gefahrenpotential besitzen, muss klar definiert werden, wie auf eine Abweichung zu reagieren ist.

Natürlich beinhaltet dieser Flowchart nur die grundlegendsten Aktivitäten. Genaueres vorgehen muss nach der Evaluation definiert werden.



## 4.4. Umsysteme

- **Prozess**

  Prozesse müssen respektiert werden.

  Vorhandene Prozesse welche im Netzwerk aktiv sind, sollen weder beeinflusst noch abgeändert werden.
- **Richtlinien**

  Implementierte Richtlinien sollen weiterhin respektiert und befolgt werden.

  Müssen eventuell erweitert werden

## 4.5. Schnittstellen

- Entscheidung und Alayse bei gemeldeten Analysen
- Kommunikation bei true positiv alerts
- Aktive anpassung der Baseline
- Kommunikation bei Änderungen im Netzwerk

## 4.6. Systemgrenzen

- Analyse betrifft nur das PoC LAN und darf nicht auf das produktive LAN ausgeweitet werden.

## 4.7. Gemeinsamkeiten

Alle Untersysteme müssen einen gewissen Grad an Compliance mit den gegebenen IST-Zuständen aufweisen können. Dies bedeutet, dass bei bereits eingesetzter Software/Hardware APIs zur verfügung stehen müssen, um effektiv Anomalien erkennen zu können.

# 5. Erhebung

## 5.1. IST-Zustand

Das POC LAN @Gitlab besitzt ein NMS (Network Monitoring System) welches bereits rudimentär den Status des Netzwerkes überwacht (Throughput). Anomalien können allerdings nicht näher erkannt werden, noch kann mit Genauigkeit gesagt werden, wo die Anomalie aufgetreten ist.

## 5.2. Quantitative Methode

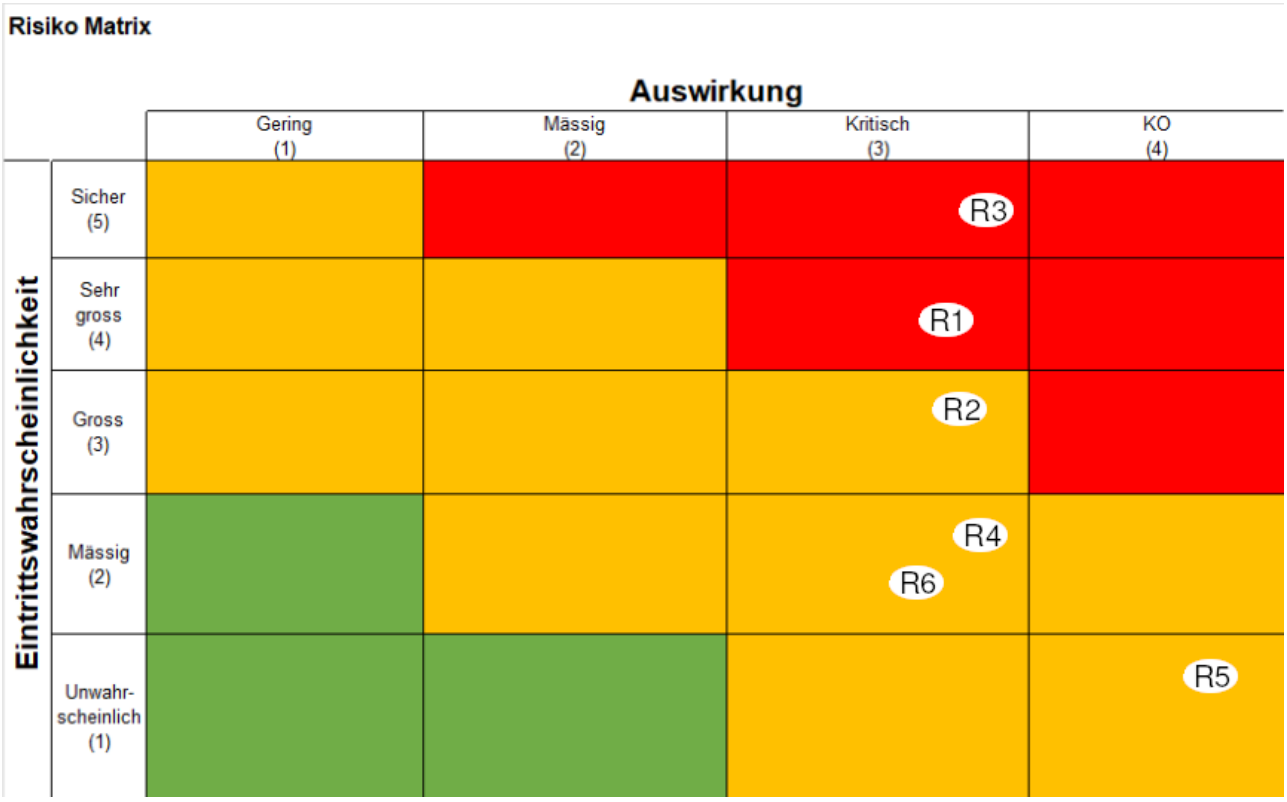
### 5.2.1. Beobachtungen, Messungen

Um einen besseren Überblick über die vorhanden Infrastruktur zu bekommen, wird mithilfe einigen Tools Messungen und Beobachtungen anhand des POC Lans durchgeführt. Die genaue Analyse der Beobachtungen wird in der Hauptstudie genauer analysiert.

## 5.3. Risiken

- [R1] "Shadow IT" führt zu Fehlkonfigurationen.
- [R2] Lateral movement wird nicht erkannt.
- [R3] Infektionen bleiben über längeren Zeitraum unerkannt.
- [R4] Rouge Systeme können im Netzwerk schaden anrichten.
- [R5] APT deployt einen persistent backdoor.
- [R6] Zugangsdaten und Informationen können geleakt werden.

### 5.3.1. Risikoanalyse



# 6. Würdigung

## 6.1. IST-Zustand

Die Infrastruktur bietet eine Basis, um Ausfälle einzelner Dienste zu sehen und die Administratoren zu informieren. Paketverlust oder andere Anomalien können zzt. noch nicht erkannt werden. Da bereits Switches, welche via SNMP überwacht werden, vorhanden sind kann darauf aufgebaut werden weitere Informationen zu bekommen.

## 6.2. SWOT-Analyse

Die SWOT-Analyse soll Aufsicht über den momentanen Zustand geben.
Mithilfe der Matrix wird erhofft zukünftige Chancen sowie momentane Schwächen besser feststellen zu können.

### 6.2.1. Rahmenbedingungen

- Firma überwachen nur ihren Perimeter, nicht aber innerhalb der Segmente.
- Durchschnittlicher lifecycle eines Databreaches beträgt 279 Tage IBM
- Erst nach 207 Tagen, wird im Durchschnitt ein Einbruch gefunden.
- Firmen setzen häufig auf AV/Endpoint Security und Firewalls. Allerdings nicht auf HIDS/NIDS

### 6.2.2. Stärken/Schwächen Profil

#### Stärken

Auch wenn innerhalb des POC LANs kein Mechanismus vorhanden ist um Anomalien zu erkennen, sind dennoch die rudimentären Anforderungen vorhanden um solch ein System einzurichten. Ebenfalls besteht bereits eine Perimeter-Überwachung welche die wichtigsten Dienste, welche ein fehlerfreies arbeiten garantiert. Da der grösste Teil der Infrastruktur auf Open Source basiert, ist es ebenfalls ein leichtes, Module zu erweitern und anzupassen.

#### Schwächen

Die Umgebung an sich besitzt ein unzureichendes Monitoring. Anomalien können nur schlecht oder gar nicht erkannt werden. Im Falle einer entdeckten Anomalie, kann aufgrund fehlender zentralen Loggings nicht garantiert werden, dass der Verursacher gefunden werden kan.

#### Chancen

Die Software für die Anomalie Erkennung soll einen Mehrwert in der gesamten Struktur des Netzwerkes erbringen. Allgemeine Risiken und Gefahren können durch ein frühzeitiges erkennen eingedämmt oder direkt unterbunden werden. Durch den Einsatz eines Log-Collectors und eines NMS, können Anhaltspunkte zur Überwachung des Netzwerks gegeben werden.

**Gefahren**

Durch die erhöhte Überwachung des Netzerkes könnte es zu Datenschutz problemen führen. Sowie erhöhter Administrativer Aufwand die Baseline zu wahren.

### 6.2.3. Matrix

| Positiv | Negativ |
|---|---|
| \| **S** \| Stärken | \| **W** \| Schwächen |
| • Know-How vorhanden<br>• Grundlegende Überwachung<br>• OSS im Einsatz<br>• Übersichtliche Infrastruktur<br>• Gute Dokumentation | • Nur Perimeter Überwachung<br>• Kein zentrales Logging<br>• Keine bekannte Baseline<br>• Keine vorgenommene Analyse |
| \| **O** \| Chancen | \| T \| Gefahren |
| • Lateral movement kann erkannt werden<br>• Detectionrate kann erhöht werden<br>• Systeme können an einen Log-Collector integriert werden<br>• "Shadow IT" kann minimiert werden | • Performance könnte schlechter werden<br>• Administratorwechsel könnte zu verwirrung führen<br>• Schlechte Dokumentation kann zu Problemen führen<br>• Netzwerk unterbruch aufgrund fehlkonfigurationen |

## 7. Lösungsvarianten

Um das bereits vorhanden POC LAN auszubauen und mit einem Anomalie Erkennungsmodul auzustatten, ist es Notwendig, einen zentralen Log-Collector, sowie ein richtig konfiguriertes NMS zu haben. Anhand der definierten Kriterien und Gewichtungen werden die Lösungen gegenübergestellt.

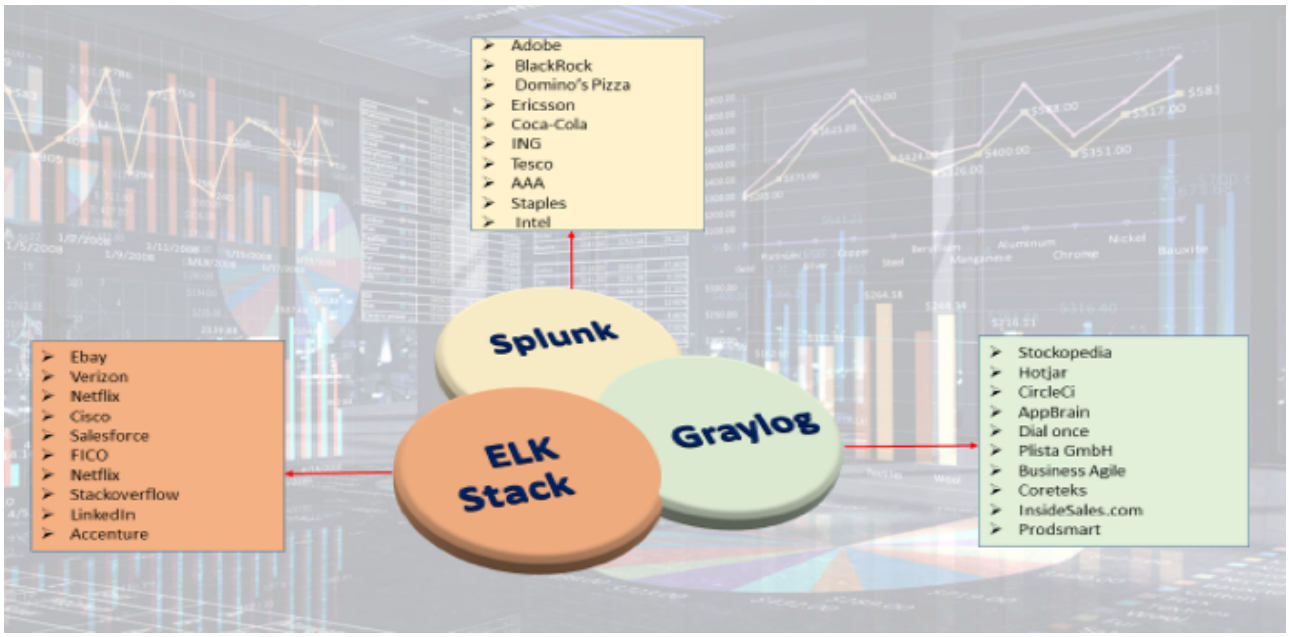
### 7.1. Log-Collector

> Log management is the collective processes and policies used to administer and facilitate the generation, transmission, analysis, storage, archiving and ultimate disposal of the large volumes of log data…
>
> — techtarget.com

Zentrales logging ist in Hinsicht auf ein Anomalie Erkennungstool beinahe unumgänglich.
Anhand der Logs können folgende Dinge erkannt werden:

- Zugriffsverletzung
- Passwortänderungen
- Neustarts
- Änderungen von Konfigurationen

Da beinahe (eine Ausnahme) nur GNU/Linux Server im Einsatz sind, muss der Log-Collector Syslog unterstützen um die Logs sammeln zu können.
Um die Software besser bewerten zu können, wird jeweils eine virtuelle Maschine mit einer Instanz der Software installiert und analysiert.



### 7.1.1. V0 | Null

Logs werden weiterhin ohne zentralen Server gemanagt. Dies würde dazu führen, dass Logs einzeln von den Devices abgeholt werden müssten.

#### Keypunkte

- Geringster initialer Aufwand.
- Höherer Aufwand in der Programmierung.
- Erhöht die Gefahr, Anomalien nicht zu erkennen.

#### Vorteile

##### Geringster initialer Aufwand

Ohne Log-Collector müsste das Syslog bei keinem Server konfiguriert werden. Somit könnte man initial den Aufwand minimieren und die Zeit in andere Punkte investieren.

#### Nachteile

##### Höherer Programmieraufwand

Die Logs können nicht mehr von einer zentralen Stelle abgefragt werden. Dies würde dazu führen, dass man eine Liste von vorhandenen Servern führen müsste und diese einzeln jeweils über ihren momentan Stand abfragen.

##### Anomalien nicht erkennen

Zentrales Log Management erhöht die durchsicht in einem Netzwerk immens. Ohne Log-Collector können einzelne Server/Dienste vergessen gehen oder nicht abgefragt werden.

### 7.1.2. V1 | Graylog

Graylog ist nicht nur eine Log Management Lösung. Sondern beinhaltet ein komplettes SIEM. Der unterliegende Storage welcher auf Elasticsearch basiert, ermöglicht eine schnelle Suche der Dateien. Besonders interessant ist das REST API und ihre OpenSource Lizenz welche Graylog kostengünstig und erweiterbar macht.


#### Keypunkte

- Open Source und Enterprise Model
- Free (Enterprise bis 5GB/Day)
- RESTful
- *all-or-nothing* solution
- Limited scope

#### Vorteile

##### Free

Das Open Source sowie Enterprise Model sind von interesse wenn es darum geht, Logs zu collecten und abzuarbeiten. Falls es zu einer Entscheidung zu Graylog kommen sollte, wird allerdings auf das Enterprise Model verzichtet, da es sich nur um ein POC handelt.

##### RESTful

Das integrierte REST API ermöglicht es, selbst Wrapper für die Applikation zu schreiben um schnell an die gewünschten Informationen zu kommen.

#### Nachteile

##### All-Or-Nothing

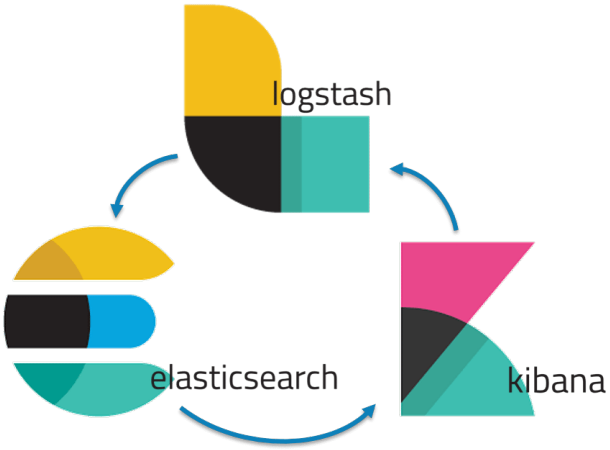
Graylog kann eines gut - Logs managen.
Andere Lösungen bieten die Funktionalität direkt Graphen aus den Logs herauszuschreiben. Graylog hingegen benötigt dafür weitere Tools (Graphana).

##### Limited scope

Wie oben genannt, bietet Graylog die möglichkeit nicht an, Logs in andere Daten umzuwandeln um sie Beispielsweise in den KPIs anzuzeigen.

### 7.1.3. V2 | ELK-Stack

Elasticsearch ist einer der Platzhirsche in Datenverarbeitung und so ziemlich alles, was mit Bigdata und Datenanalyse zu tun hat. Durch Kibana besitzt es eine relativ gute Weboberflächte und mit Logstash besitz es die Möglichkeit Logs zu empfangen und zu bearbeiten.




**Keypunkte**

- Vollumfänglich
- Clients senden via Beats Logs zum Server
- RESTful
- Hohe Lernkurve
- Hohe Wartungskosten
- Logtransformierung
- Keine Authentifizierung in der Free-Version
- Aufteilung via Shards

**Vorteile**

**Vollumfänglich**

Der ELK-Stack beinhaltet alles was man zum Log Managen braucht. Via Beats können Logs von Windows sowie Linux Client empfangen und transformiert werden. Kibana ermöglicht das Anzeigen der Logs in Realtime direkt in einer Weboberfläche, und Elasticsearch bietet eine vollumfängliche analytische Umgebung um Logs zu analysieren an.

**Beats**

Dadurch das eine Software genutzt wird um Logs aus den Clients zum Server zu senden (würde auch via Syslog gehen), können zusätzliche Informationen angehängt werden. Zum Beispiel ermöglicht dies dem Server mitzuteilen, welche Beats Version momentan auf dem Client läuft.

**RESTful**

Das integrierte REST API ermöglicht es, selbst Wrapper für die Applikation zu schreiben um schnell an die gewünschten Informationen zu kommen.

**Nachteile**

**Hohe Wartungskosten**

Die Erfahrung mit ELK zeigt, dass die Kosten der Wartung des Servers ziemlich hoch sind (Zeitkosten). Falls ein Shard failen sollte, ist eine rückführung der korruptierten Logs extrem aufwändig bis beinahe unmöglich. Dazu kommt, dass Beats nicht mit allen Logtypen umgehen kann und unter gewissen umständen eigene Parser geschrieben werden müssen.

**Hohe Lernkurve**

Die Administration eines ELK-Stack ist nicht einfacht. Bereits das Verwalten von mehreren Nodes ist extrem Zeitaufwändig, geschweige von den Konfigurationsmöglichkeiten welche Logstash mit sich bringt.

**Keine Authentifizierung**

Die Community Version des ELK-Stacks (Kibana) bietet keine Authentifizierung an. Das heisst, das zusätzlich ein Apache2/nginx oder ein reverse Proxy vorgeschalten werden müsste, welcher die Authentifizierung übernimmt.

### 7.1.4. V3 | Splunk

Splunk sollte jedem welcher sich mit Log Management etwas beschäftigt hat ein Begriff sein. Besonders interessant ist Splunk Enterprise Security (ES) welches als SIEM von Splunk selbst dient. Es bietet eine weite Modulare erweiterbarkeit durch Community erstellte Apps dar.


#### Keypunkte

- Eigene Query Sprache (SPL)
- Standard APIs
- Easy Setup
- Monitoring und Alerting
- Kein Java
- Properitär

#### Vorteile

##### Standard APIs

Wie bei den restlichen Tools wird ein RESTful sowie HTTP API verwendet.

##### Monitoring und Alerting

Splunk besitzt bereits selbst die Funktionalität, Daten zu monitoren und bei Sonderfällen die Administratoren zu informieren.

##### Easy Setup

Das Setup benötigt nur einen Splunk Server und evtl. einen Splunk Forwarder. Zusätzliche Komponenten wie bei den anderen Lösungen werden nicht benötigt.

##### Kein Java

Es lebt sich leichter mit Server welche kein Java benötigten. Splunk ist in C++ / Python geschrieben und benötigt dadurch keine JVM oder zusätzliche Software.

#### Nachteile

##### Properitär

Splunk ist nicht Open Source. Dies behält den Nachteil, dass man nicht sicher gehen kann, was die Software genau unterliegend macht. Im Fehlerfall, kann nicht direkt im Code nachgeschaut werden, was nun schief gelaufen ist.

### 7.1.5. V-Bewertung

| Ausscheidekriterien / Mussziele | | Lösungsvariante 1 | | | Lösungsvariante 2 | | | Lösungsvariante 3 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Graylog | | | ELK-Stack | | | Splunk | | |
| **Optimierungskriterien** | **Gewicht** | **Info** | **Punkte** | **Produkt** | **Info** | **Punkte** | **Produkt** | **Info** | **Punkte** | **Produkt** |
| API | 28.57 | HTTP/REST | 10 | 285.71 | HTTP/REST | 10 | 285.71 | HTTP/REST | 10 | 285.71 |
| Dokumentation | 23.81 | | 10 | 238.10 | | 7 | 200.00 | | 10 | 285.71 |
| Performance | 9.52 | | 8 | 76.19 | | 10 | 285.71 | | 9 | 257.14 |
| Skalierbarkeit | 4.76 | | 6 | 28.57 | | 10 | 285.71 | | 8 | 228.57 |
| Lizenz | 0.00 | | 8 | 0.00 | | 4 | 114.28 | | 8 | 228.57 |
| Ausgereift | 14.29 | | 8 | 114.29 | | 8 | 228.57 | | 10 | 285.71 |
| Aktivität | 19.05 | | 10 | 190.48 | | 5 | 142.86 | | 10 | 285.71 |
| **Summe** | **100.00** | | | 933.34 | | | 1542.83 | | | 1857.12 |
| Zielereichungsgrad | | | 21.54% | | | 35.60% | | | 42.86% | |
| **Rang** | | | | 3 | | | 2 | | | 1 |

**Analyse**

Grundsätzlich war es ein sehr enges Rennen.
Bei einem ELK-Stack ist die Aktivität etwas ein Problem. Viele Tools laufen nur mit älteren Versionen von Elasticsearch und sind daher auf bestimmte Konfigurationen angewiesen.
Graylog hingegen würde ich gerne anderswo nochmals testen. Grundsätzlich kann man allerdings sagen, das Splunk eines der weitverbreitesten und innovativsten Log Monitoring Managements ist. Daher viel die wahl u.a. der grossen Community und ausführliche Dokumentation auf Splunk.

## 7.2. Network Monitoring System

TODO

### 7.2.1. V0 | Null

TODO

### 7.2.2. V1 | PRTG

TODO

### 7.2.3. V2 | Nagios

TODO

### 7.2.4. SW-Bewertung

### 7.2.5. V-Bewertung

# 8. Weiteres Vorgehen

## 9. Freigabe

## 10. Darstellungsverzeichnis

## 11. Glossar