



Frankfurt University of Applied Sciences
Fachbereich 2 - Studiengang Informatik

Abschlussarbeit

zur Erlangung des akademischen Grades

Bachelor of Science (B.Sc.)

Entwicklung einer global verteilten Infrastruktur zur Speicherung, Strukturierung und Analyse von Logdaten

Vorgelegt von	Daniel Brandtner
Matrikelnummer	987990
am	10. September 2014
Referent	Prof. Dr. Christian Baun
Korreferent	Alexander Mützel

Eidesstattliche Erklärung

Hiermit erkläre ich,

- dass ich die vorliegende Abschlussarbeit selbstständig angefertigt habe,
- dass ich nur die angegebenen Quellen genutzt habe,
- dass ich aus fremden Arbeiten übernommene Stellen kenntlich gemacht habe,
- dass ich keine unerlaubte fremde Hilfe in Anspruch genommen habe.

Frankfurt am Main, 10. September 2014

Daniel Brandtner

Danksagung

Für die fortwährende und großartige Unterstützung in meinem Studium möchte ich mich herzlich bei meiner Familie, sowie bei Andreas Seeber und Alexander Mützel bedanken.

Prof. Dr. Christian Baun danke ich für die Betreuung dieser Abschlussarbeit. Außerdem auch allen Freunden und Bekannten, die sich die Zeit genommen haben, sie gegenzulesen und mir Feedback zukommen zu lassen.

Den Mitarbeitern von DENIC, im Besonderen Christian Stein danke ich für die freundliche Unterstützung.

Zusammenfassung

Das Erheben und Verarbeiten von Systemprotokollen (Logdaten) ist ein elementarer Aspekt jeglicher Systemadministration. Ein sicherer und performanter Betrieb von informationstechnischen Anlagen hängt davon ab, ob Logdaten in angemessener Menge und mit sinnvollem Fokus zur Verfügung stehen.

Das System-Logging auf Einzelsystemen oder in kleinen Netzwerken kann in der Regel manuell eingerichtet und ausgewertet werden. Es besteht dabei kein Bedarf an Spezialsoftware, die über die typischen, von den Betriebssystemen bereitgestellten Lösungen hinaus geht. Mit steigender Anzahl an relevanten Systemen, beispielsweise in Unternehmensnetzwerken, ist die händische Verarbeitung von Logdaten allerdings nicht mehr unter vertretbarem Aufwand möglich.

Diese Arbeit behandelt die Fragestellung, wie Logdaten von global verteilten Systemen zentralisiert erfasst, strukturiert und bei Bedarf analysiert werden können. Dies geschieht in Kooperation mit der deutschen Internet-Registrierungsstelle DENIC, die ein weltumspannendes Informationsnetzwerk betreibt und somit Bedarf an einer intelligenten Lösung zur Verwaltung von Logdaten hat. Hierfür soll unter Einsatz der professionellen Logging-Software „Splunk Enterprise“ eine einheitliche, über die einzelnen Standortgrenzen hinweg arbeitende Logging-Infrastruktur entwickelt werden.

Inhaltsverzeichnis

1	Einführung des Themas	1
1.1	Motivation	2
1.2	Zielsetzung	2
1.3	Aufbau der Arbeit	3
2	Logging im Linux-Umfeld	4
2.1	Logdaten allgemein	4
2.2	Das syslog-Protokoll	6
2.3	Rotation	11
2.4	Kompression	13
2.5	Schwierigkeiten der typischen Methoden	15
2.5.1	Abweichende Syntax	15
2.5.2	Wachstum von Logdateien	16
2.5.3	Teure Negativsuche	16
2.5.4	Verschiedenartige Werkzeuge	17
2.5.5	Zugriff auf Log-Journals	17
2.5.6	Nicht-unixoide Betriebssysteme	18
3	Verbesserungspotential durch Splunk Enterprise	19
3.1	Gleichartigkeit der Daten	19
3.2	Effizientes Durchsuchen	20

3.3	Zentrale Speicherung	21
3.4	Langfristige Speicherung	21
3.5	Analyse und Aufbereitung	21
3.6	Search Processing Language (SPL)	22
3.7	Systemunabhängigkeit	22
4	Komponenten von Splunk	23
4.1	Indexer	24
4.2	Forwarder	25
4.2.1	Universal Forwarder	26
4.2.2	Heavy Forwarder	26
4.3	Search Heads	28
4.4	Master Nodes	29
5	Splunk-interne Strukturierung der Daten	30
5.1	Parsing	31
5.2	Indexierung	32
5.3	Buckets	33
6	Typische Einsatzszenarien	36
6.1	Einzelner Splunk-Server	36
6.2	Mehrere Indexer	38
6.3	Indexer pro Standort mit einem zentralen Search Head	40
6.4	Search Head Pooling und Indexer-Cluster	42
7	Implementierung der Splunk-Infrastruktur	43
7.1	Bestehende Lösungen am zentralen Standort	43
7.2	Bestehende Lösung an den verteilten Standorten	44
7.3	Zielbild	46
7.4	Implementierung am zentralen Standort	47

7.4.1	Konfiguration des Master Node	47
7.4.2	Konfiguration der Indexer	49
7.4.3	Konfiguration der Forwarder	52
7.4.4	Sonderfall: Netzwerkhardware	54
7.4.5	Konfiguration des Search Heads	56
7.5	Konfiguration cluster-lokaler Datenquellen	58
7.6	Konfiguration der Lizenzen	60
7.7	Integration der entfernten Standorte	62
8	Anwendungsbeispiele	64
8.1	Suchen	64
8.2	Reports und Dashboards	67
9	Sicherheit	70
9.1	Verfügbarkeit	70
9.2	Authentizität	71
9.3	Vertraulichkeit	72
9.4	Integrität	73
10	Lizenzierung	74
10.1	Lizenzverletzung	75
11	Fazit	77
11.1	Vergleich der Plattformen	78
11.2	Ausblick	78
	Abbildungsverzeichnis	79
	Tabellenverzeichnis	81
	Literaturverzeichnis	82

Kapitel 1

Einführung des Themas

Die DENIC eG¹ in Frankfurt am Main ist die deutsche Internet-Registrierungsstelle (Registry) und somit zuständig für den Betrieb der Internet-Toplevel-Domain *.de*. Dies bedeutet, den Teil des globalen Domain Name System (DNS) zu betreuen, welcher Domain-Namen unterhalb von *.de* in numerische Adressen (IP-Adressen) übersetzt. Außerdem werden Registrierungs- und Auskunftsdienste bereit gestellt, mit denen *.de*-Domains registriert, gelöscht oder auf ihre Eigentümerinformationen hin abgefragt werden können.

Diese Registrierungs- und Auskunftsdienste werden von einem zentralen Standort, der „Registry Service Location“ (RSL), aus angeboten. Hier erbringen mehrere Hundert Serversysteme ihre jeweiligen Dienstleistungen. Das wichtigste Angebot, die Zuordnung von Domain-Namen zu IP-Adressen, wird jedoch nicht von einem zentralen Standort aus angeboten sondern von mehreren Dutzend global verteilten, kleineren Standorten. Diese heißen „Name Service Locations“ (NSLs) und enthalten jeweils einen standardisierten Verbund von etwa einem Dutzend Serversystemen. Gemeinsam betrachtet bilden RSL und NSLs also ein großes Unternehmensnetzwerk, in dem die Überwachung aller einzelnen Systeme aufwändig ist. Daher sind spezielle Methoden gefordert um trotz der Größe des Netzwerks eine effiziente Logdatenanalyse zu ermöglichen.

¹<http://www.denic.de>

1.1 Motivation

Sowohl in der RSL als auch in den NSLs kommen hauptsächlich Linux-Systeme zum Einsatz. Die typischen Linux-Bordmittel zur Protokollierung von Systemereignissen stellen sich unter wachsenden Anforderungen aber als unflexibel heraus.

Sie sind beispielsweise von einem hohen Wartungs- und Speicherplatzbedarf geprägt. Zum Einsparen von Speicherplatz ist eine rudimentäre Kompression der Logdaten auf Dateibasis möglich, allerdings geschieht dies direkt auf Kosten von Zugriffsmöglichkeiten und Auswertbarkeit der gesammelten Daten.

Auch die Möglichkeiten zur Analyse und Darstellung der Logdaten mit Linux-Bordmitteln sind eher rudimentär. Die benötigten Werkzeuge, um rohe Logdaten in eine aufbereitete Form zu bringen, müssen meist selbst entwickelt oder angepasst werden.

Außerdem besteht in der Regel keine Möglichkeit, eine Indexierung der Logdaten vorzunehmen. Es ist daher meist nicht möglich, auf effiziente Verfahren zum Durchsuchen großer Datenbestände zurückzugreifen.

1.2 Zielsetzung

Inhalt dieser Arbeit ist es, sowohl für die Systeme des zentralen Standorts als auch für die global verteilten Standorte eine gemeinsame und homogene Logging-Infrastruktur zu entwickeln.

Sie soll beim Konsolidieren der einzelnen Meldungen aller Systeme vom aktuellen Stand der Technik profitieren. Besonderes Augenmerk liegt hierbei auf der Zuverlässigkeit der Infrastruktur und einer intelligenten Reduzierung des benötigten Speicherplatzbedarfs. Außerdem soll eine Indexierung der anfallenden Logdaten umfassende Auswertungen mit möglichst geringerem Zeitbedarf ermöglichen. Dies ist wünschenswert, da effiziente Analysen der vorhandenen Logdaten im Fehlerfall schnelle Fehlerbehebung oder Optimierungen ermöglichen.

Zur Umsetzung dieses Vorhabens möchte DENIC die proprietäre Softwarelösung „Splunk Enterprise“² einsetzen. Die vorliegende Arbeit soll prüfen, ob das Produkt den Vorstellungen und Ansprüchen des Unternehmens gerecht werden kann. Weiterhin soll die Implementierung der Software unter den unterschiedlichen Gegebenheiten von zentralem und verteilten Standorten beschrieben werden.

1.3 Aufbau der Arbeit

Zunächst werden einige typische Verfahren und Werkzeuge zum Erheben von Logdaten im Linux-Umfeld beschrieben, um die klassischen Methodiken zu veranschaulichen. Im darauf folgenden Kapitel wird betrachtet, auf welche Weise die Implementierung von Splunk die dabei herausgestellten Schwierigkeiten überwinden kann.

Anschließend wird näher auf die oberflächlichen und internen Funktionen von Splunk eingegangen. Es werden die einzelnen Komponenten der Software vorgestellt und die zum Einsatz kommenden Verfahren zur Strukturierung der gesammelten Logdaten näher betrachtet.

Anhand von typischen Implementierungen von Splunk-Infrastrukturen wird eine mögliche Variante für die Standorte des Unternehmens vorgeschlagen.

Im darauf folgenden Kapitel werden einige Anwendungsbeispiele und Funktionen von Splunk vorgestellt.

Anschließend wird betrachtet, auf welchen Wegen die Ausfallsicherheit, Zuverlässigkeit und Vertraulichkeit der Infrastruktur gewährleistet werden kann.

Es folgt ein abschließendes Fazit. Dabei wird auch die Frage behandelt, wie in Zukunft weiter wachsende Anforderungen von der neuen Infrastruktur bedient werden können.

²<http://www.splunk.com>

Kapitel 2

Logging im Linux-Umfeld

Dieses Kapitel gibt eine Einführung über Logdaten als solches und beschreibt typische Verfahren zur Verwaltung dieser Daten in Linux-Umgebungen. Die dafür eingesetzten Werkzeuge können sich zwar unter den einzelnen Linux-Distributionen unterscheiden, allerdings gehören die im Folgenden erwähnten Werkzeuge zur Standardausstattung der verbreiteten Linux-Distributionen.

2.1 Logdaten allgemein

Unter Logdaten versteht man alle Arten von Aufzeichnungen über eingetretene Ereignisse oder Zustände eines Systems. Dies beinhaltet auch Fehlermeldungen oder Leistungswerte eines Systems. Die Daten können allgemeiner Natur sein und vom System selbst erzeugt werden oder von einzelnen Anwendungen, wie Web- oder E-Maildiensten, stammen.

In aller Regel werden Ereignisse zeilenweise in Textdateien festgehalten, wobei pro Zeile ein Ereignis geschrieben wird. Für gewöhnlich gibt ein Zeitstempel am Anfang jeder Zeile an, wann das Ereignis eingetreten ist beziehungsweise protokolliert wurde. Der Rest der Zeile enthält die eigentlichen Informationen über das Ereignis. Je nach Anwendung sind diese nach festen Mustern strukturiert.

Ein Ereignis kann auch mehrere Zeilen in einer Logdatei beanspruchen. Dies macht die Handhabung der Logdatei allerdings schwieriger in Bezug auf Durchsuchbarkeit oder automatisierte Erkennung von einzelnen Informationen des jeweiligen Ereignisses.

Zur Verdeutlichung folgt eine Beispielzeile aus der Logdatei eines Webservers. Ihr Format entspricht dem „Combined Logfile Format“³ (CLF), das von unterschiedlichen Webservern genutzt wird.

```
1 192.109.234.216 - - [21/Aug/2014:01:33:24 +0200] "GET /index.html
  HTTP/1.1" 200 1468 "http://frankfurt-university.de/" "Mozilla
  /5.0 (X11; Ubuntu; Linux x86; rv:31.0) Gecko/20100101 Firefox
  /31.0"
```

Listing 2.1: Logbeispiel: Combined Logfile Format

Dieses geloggte Ereignis protokolliert den Abruf einer Webseite. Die Zeile ist aus verschiedenen Informationsfeldern aufgebaut. Alle vom Webserver geschriebenen Zeilen dieser Art werden ebenfalls diesem Format folgen.

Von links nach rechts gelesen ergeben sich folgende Informationen aus der Logzeile:

192.109.234.216 IP-Adresse des aufrufenden Rechners.

[21/Aug/2014:01:33:24 +0200] Datum und Uhrzeit des Aufrufs. Die Zeitzone des Server ist gegenüber der Greenwich Mean Time (GMT) um zwei Stunden nach vorne versetzt.

GET /index.html HTTP/1.1 Der Aufruf nutzte das für Webseiten standardmäßige „HyperText Transfer Protocol“ (HTTP), Version 1.1, und forderte mittels dessen GET-Methode eine Seite namens `index.html` an.

200 An dieser Stelle steht der Antwortstatus des Webservers auf die HTTP-Anfrage codiert. 200 bedeutet, dass der Aufruf erfolgreich bedient wurde.

³<https://httpd.apache.org/docs/1.3/logs.html#accesslog>

1468 Größe der gesendeten Antwort in Byte.

http://frankfurt-university.de/ Falls vorhanden, Adresse der Webseite, die auf die lokale `index.html` verwiesen hat.

(Rest der Zeile) Hier kann der sogenannte „UserAgent“ stehen. Er enthält, sofern vom aufrufenden System mit gesendet, weitere Angaben über den aufrufenden Rechner. Dies sind unter Anderem die eingesetzte Browser-Version und Art des Betriebssystems.

Dieses Beispiel eines Webseitenaufrufs steht exemplarisch für andere Anwendungen. Logzeilen jeglicher Anwendungen zeichnen sich dadurch aus, dass sie festen Formaten folgen und somit automatisiert durchsuchbar sind.

2.2 Das syslog-Protokoll

Eine Anwendung muss den Schreibzugriff auf ihre Logdateien nicht selbst verwalten. Die generierten Logdaten können beispielsweise an einen lokalen syslog-Dienst übergeben werden, welcher sie verarbeitet oder über ein Netzwerk an syslog-Dienste auf anderen Systemen übermittelt. Die Übermittlung erfolgt dabei mittels des standardisierten syslog-Netzwerkprotokolls. Lokale Verarbeitung durch einen syslog-Dienst kann bedeuten, die Daten in Echtzeit auf einem Bildschirm auszugeben, per E-Mail an einen Benutzer zu versenden oder sie, im typischsten Fall, in Logdateien niederzuschreiben. [1]

Die heute verbreitetsten Implementierungen des syslog-Protokolls unter Linux sind die ursprüngliche Referenzimplementierung des Protokolls, „syslogd“[1], außerdem „syslog-ng“⁴ und das neuere „rsyslog“⁵. Diese Dienste laufen kontinuierlich im Hintergrund eines Systems und implementieren Empfang oder Weiterleitung von Meldungen gemäß dem syslog-Protokoll.

⁴<http://www.balabit.com/network-security/syslog-ng/>

⁵<http://www.rsyslog.com>

Da Bildschirme in der Darstellungsfläche beschränkt sind und massenhafte E-Mails zu Problemen anderer Arten führen können, sind diese Methoden nur für kleine Mengen von Logdaten praktikabel.

Das Anlegen und Vorhalten in Logdateien ist eine bessere Methode, um auch zu späteren Zeitpunkten auf große Mengen von Logdaten zugreifen zu können.

Der „Filesystem Hierarchy Standard“ (FHS) ist eine Richtlinienammlung zur Nutzung der Verzeichnisstruktur von UNIX-ähnlichen Betriebssystemen. Er schlägt vor, dass sämtliche erzeugten Logdaten im Verzeichnis `/var/log` oder einem Unterverzeichnis davon abgespeichert werden. [2] Hier befinden sich demnach Logdateien von einzelnen Anwendungen sowie anwendungsunabhängige Dateien, die vom System selbst geführt werden. Anwendungen, die keine eigene Logdatei führen, teilen sich in der Regel gemeinsame. Die drei vorher genannten syslog-Implementierungen führen jeweils eigene allgemeine Logdateien, beispielsweise `/var/log/messages` oder `/var/log/syslog`.

Beispiele für Logdateien von bestimmten Anwendungen sind Zugriffsprotokolle von Webservern (siehe Beispiel in Listing 2.3), Zustellungsprotokolle von E-Mailservern oder auch die Protokollierung von Benutzeranmeldungen am System.

Eine Anwendung ist nicht an eine einzelne Logdatei gebunden sondern kann so viele Dateien beanspruchen, wie es dem Entwickler oder dem Benutzer sinnvoll erscheint. Auch kann die selbe Meldung in mehrere Logdateien geschrieben werden. Beispielsweise könnte ein Mailserver seine Protokolldaten getrennt nach ein- und ausgehenden E-Mails in separate Dateien schreiben. Er könnte auch eine separierte Protokollierung von virenbefallenen E-Mails oder unerwünschten SPAM-Mails führen. Die Virenfunde könnten zusätzlich in einer systemweiten Sicherheitslogdatei festgehalten werden. Bei einem Webserver erscheint es hingegen naheliegend, die Logdateien jeweils pro angebotener Webpräsenz zu strukturieren. Auch die Protokollierung von Benutzeranmeldungen am System könnte beispielsweise in separate Logdateien für erfolgreiche und fehlgeschlagene Anmeldevorgänge aufgeteilt werden.

Für gewöhnlich kann eine Anwendung durch ihre eigene Konfiguration veranlasst werden, bestimmte Logdateien zu befüllen. Alternativ kann sie ihre Daten an den syslog-Dienst liefern, welcher die Weiterleitung zu den passenden Logdateien übernimmt. Sofern alle Anwendungen eines Systems das Weiterreichen an den syslog-Dienst unterstützen, entsteht eine zentrale Konfigurationsmöglichkeit für das gesamte Logging-Geschehen des Systems.

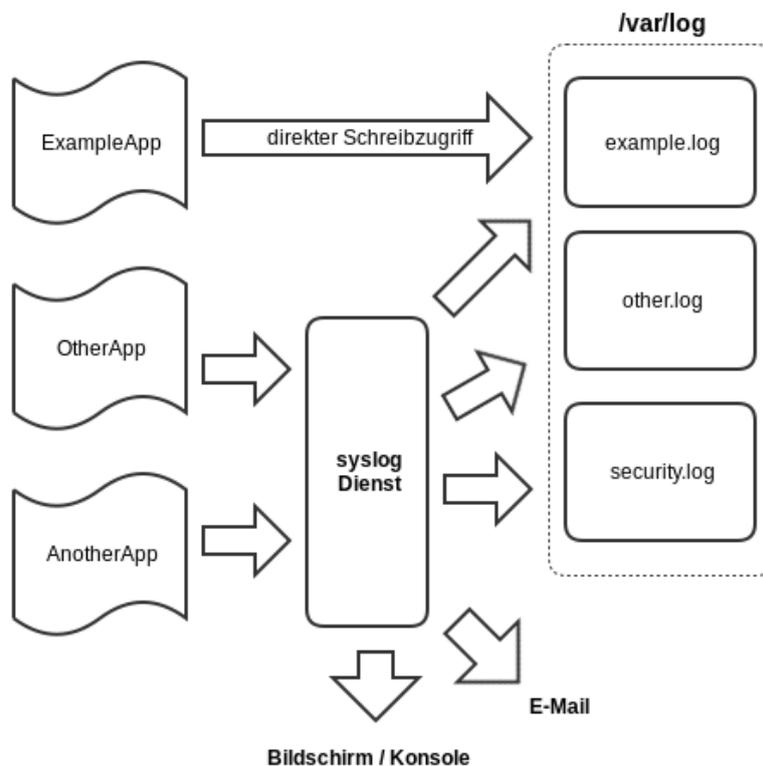


Abbildung 2.1: syslog gegenüber direktem Schreiben

Neben dem Fall, dass eine Anwendung in mehrere Logdateien schreibt, kann es auch sinnvoll sein, die Meldungen mehrerer Anwendungen in der selben Logdatei zu sammeln. So können beispielsweise lokale Anmeldungen am System und entfernte Anmeldungen über Netzwerkprotokolle wie SSH gemeinsam in der selben Logdatei protokolliert werden, obwohl die Vorgänge nicht von den selben Prozessen bedient werden.

Um eine Gruppierung von thematisch gleichen Meldungen zu ermöglichen, bietet syslog seit RFC 3164 ⁶ sogenannte „facilities“ an. Hierbei handelt es sich um definierte „Kanäle“, die anstelle der einzelnen Anwendungen den Logdateien zugeordnet werden. Anstatt für jede ähnliche Anwendung einzeln das Schreiben in die selbe Logdatei zu definieren, reicht es aus, die Anwendungen in die gleiche facility schreiben zu lassen. [3]

Tabelle 2.1: Einige facilities des syslog-Standards

Nummer	Name	Zweck
0	kern	Kernel-Meldungen
1	user	Meldungen von Anwendungen
2	mail	E-Mail-Dienste
3	daemon	Meldungen von sonstigen Diensten
4	auth	Sicherheitsmeldungen
5	syslog	syslog-eigene Meldungen
6	lpr	Drucker-Dienste
...
11	ftp	FTP-Dienste
...
16 - 23	local _N	frei nutzbar

Anhand dieser Tabelle wird ersichtlich, nach welchem Prinzip die facilities definiert sind. Beispielsweise können die verschiedenen Dienste für Benutzeranmeldungen, die sich eine Logdatei namens `auth.log` teilen werden, nun gemeinsam in die „auth-facility“ schreiben. syslog kümmert sich um die Zuordnung von „auth-facility“ zur Datei `auth.log`. Es muss nicht mehr das Logging jeder einzelnen Anwendung auf die besagte Datei hin konfiguriert werden.

Zusätzlich zu den facilities, die eine thematische Gruppierung ermöglichen, sieht das syslog-Protokoll seit RFC 5424 ⁷ auch eine Markierung der Relevanz einer jeden Meldung vor. Hierzu definiert der RFC „severity levels“, die absteigend von „7“ bis „0“ ansteigende Kritikalität einer Meldung signalisieren. [4]

⁶<http://www.ietf.org/rfc/rfc3164.txt>

⁷<http://www.ietf.org/rfc/rfc5424.txt>

Tabelle 2.2: Severity levels des syslog-Standards

Grad	Name	Bedeutung
7	debug	Statusmeldungen für Entwickler einer Applikation
6	info	Normale Betriebsmeldungen für Berichte und Messungen
5	notice	Leichte Abweichungen vom Normalbetrieb, aber keine Fehler
4	warn	Abweichung vom Normalbetrieb, die in absehbarer Zeit eine Aktion erfordert
3	err	Fehler, der in absehbarer Zeit eine Aktion erfordert
2	crit	Fehler, der sofort eine Aktion erfordert, aber keine oder geringe Auswirkungen auf den Betrieb hat
1	alert	Fehler, der sofort eine Aktion erfordert, und direkte Auswirkungen auf den Betrieb hat
0	emerg	Schwerer Fehler, Ausfallsituation

Aus der Kombination von facility und severity level ergeben sich weitere Unterscheidungsmöglichkeiten für syslog. Es bietet sich beispielsweise an, thematisch gleiche, aber in ihrer Schwere unterschiedliche Meldungen in unterschiedliche Logdateien schreiben zu lassen. Ein anderer denkbarer Fall wäre, unkritische Meldungen einer Anwendungsgruppe in die eine Logdatei schreiben zu lassen, kritische Meldungen dieser Gruppe in eine andere und diese zusätzlich per E-Mail an einen Systemadministrator zu versenden.

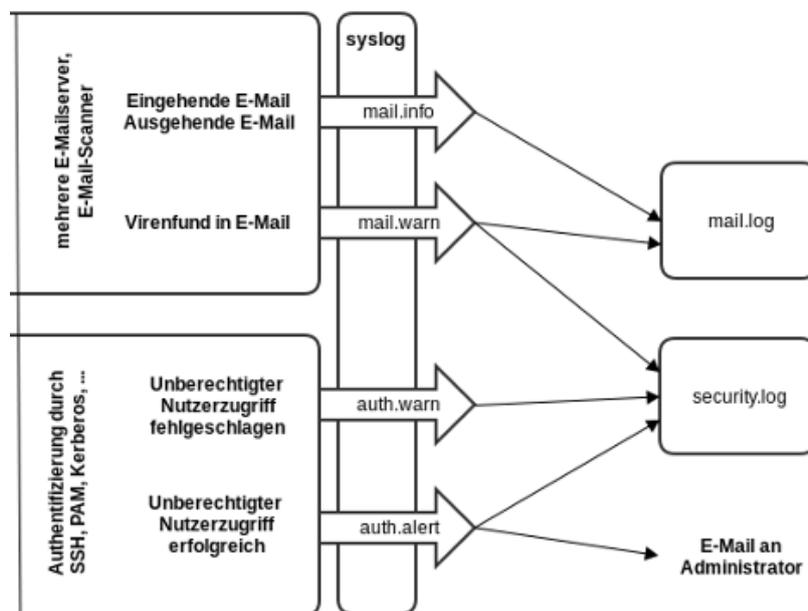


Abbildung 2.2: Anwendungsbeispiel syslog-facilities

2.3 Rotation

Durch den Einsatz des syslog-Protokolls und seiner Implementierungen ist es möglich, Ereignisse in Netzwerken und Infrastrukturen, gegebenenfalls zentralisiert, zu protokollieren. Da die überwachten Systeme aber sehr unterschiedliche Mengen von Logdaten produzieren können oder deren Volumen zeitweise stark ansteigen kann, muss der für Logfiles zur Verfügung stehende Speicherplatz mit Bedacht kalkuliert sein. Dies gilt gleichermaßen für lokales Vorhalten der Logdateien als auch in einem Szenario mit zentralisierter Datensammlung.

Durch sogenannte Rotation von Logdateien lässt sich ein Kompromiss zwischen Vorhaltezeit, Entsorgung in chronologischer Reihenfolge und Speicherplatzverbrauch erzielen. Das Vorgehen beruht auf der Annahme, dass neue Logdaten die höchste Relevanz besitzen und mit zunehmendem Alter unwichtiger werden.

Im Tages- oder Wochenrhythmus, oder anhand einer maximalen Dateigröße werden die jeweils aktuellen Logdateien als älter markiert und neue, leere Logdateien bereitgestellt, um aktuelle Logdaten aufzunehmen. Dieses Verfahren wiederholt sich so lange, bis eine definierte Anzahl an „historischen“ Logdateien vorliegt. [3] Dann wird die jeweils älteste Datei der Reihe beim nächsten Rotationsschritt entweder gelöscht, auf einen Langzeitspeicher verlagert oder per E-Mail versendet.

Zusätzlich kann es sinnvoll sein, alle Logdateien ab einer bestimmten Rotationsstufe zu komprimieren. Unter Linux verbreitete Kompressionsverfahren wie `gzip`⁸, `bzip2`⁹ oder `xz`¹⁰ können den Speicherplatzbedarf von Klartextdateien um über 99,9% reduzieren (siehe Tabelle 2.3). Dies hat allerdings zur Folge, dass Such- und Leseoperationen nun nicht mehr auf Klartextdaten operieren können und erst eine Dekomprimierung stattfinden muss. Somit werden die Zugriffe zeit- und rechenintensiver. Wann und ob eine Komprimierung von rotierten Logdateien sinnvoll ist, hängt also von den jeweiligen Anwendungsfällen ab.

⁸<http://www.gzip.org>

⁹<http://www.bzip.org>

¹⁰<http://tukaani.org/xz/format.html>

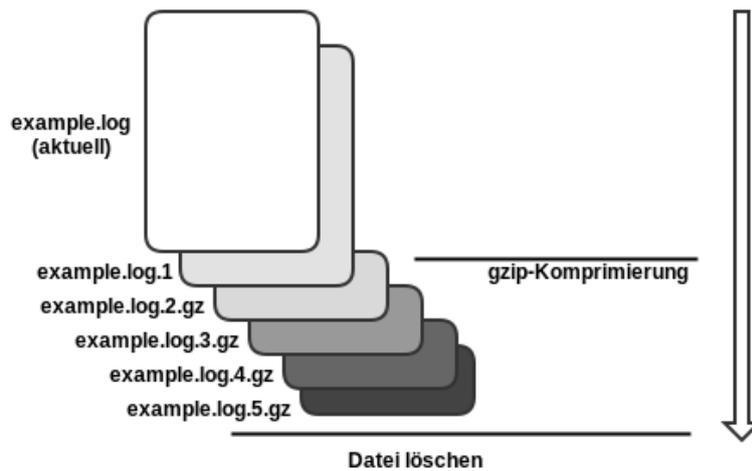


Abbildung 2.3: Rotation von Textdateien

Die Software `logrotate`¹¹ ist die unter Linux-Systemen typische Implementierung des Rotationsverfahrens. Sie erhält Konfigurationen bezüglich einzelner Logdateien oder ganzer Verzeichnisse und führt die Rotationen in den definierten Zeitabständen aus.

```
1 /var/log/messages
2 {
3     rotate 4
4     weekly
5     missingok
6     notifempty
7     compress
8     delaycompress
9     sharedscripts
10    postrotate
11        reload rsyslog >/dev/null 2>&1 || true
12    endscript
13 }
```

Listing 2.2: logrotate: Konfigurationsbeispiel

Diese Konfiguration würde veranlassen, dass `/var/log/messages` einmal wöchentlich rotiert wird. Ab der zweiten Rotation würde die Datei gzip-komprimiert und nach der vierten gelöscht. Nach jeder Rotation wird der `rsyslog`-Dienst neu geladen, um auf die ursprüngliche Datei verweisende Dateisystemzeiger zu schließen.

¹¹<http://www.fedorahosted.org/logrotate>

2.4 Kompression

Zwischen den verschiedenen etablierten Kompressionsverfahren bestehen Unterschiede in Zeitbedarf und Kompressionsraten. Im Folgenden wird eine Beispieldatei mit verschiedenen Verfahren komprimiert.

Als Ausgangsmaterial dient eine Logdatei, welche Einträge über sicherheitsrelevante Ereignisse eines Systems enthält. Die Beispieldatei belegt in Klartextform 1 GB Festplattenspeicher und enthält 10.944.140 einzelne Textzeilen.

Die Kompression erfolgt beispielhaft mit den Verfahren `gzip`, `bzip2` und `xz`. Der Test wird auf einem System mit Intel Core2Duo T5870-Prozessor mit zwei 2,0-GHz-Kernen durchgeführt. Arbeitsspeicher kann hierbei vernachlässigt werden, da die getesteten Algorithmen hauptsächlich prozessorlastig arbeiten.

Es folgt ein Auszug der Beispieldatei, an dem die Struktur der enthaltenen Einträge ersichtlich wird. Da verschiedene Prozesse in die Datei schreiben, ist das Format nicht vollständig einheitlich, aber dennoch gleichartig genug, um hohe Kompressionsraten zu ermöglichen.

```

1 ...
2 Aug 10 06:51:29 localhost sudo: nagios : TTY=unknown ; PWD=/ ;
  USER=root ; COMMAND=/usr/lib/nagios/plugins/check_ecryptfs
3 Aug 10 06:51:29 localhost sudo: pam_unix(sudo:session): session
  opened user root by (uid=0)
4 Aug 10 06:51:29 localhost sudo: pam_unix(sudo:session): session
  closed user root
5 Aug 19 18:48:37 localhost sudo: daniel : TTY=pts/0 ; PWD=/home/
  daniel ; USER=root ; COMMAND=/usr/bin/apt-get update
6 Aug 19 18:48:37 localhost sudo: pam_unix(sudo:session): session
  opened for user root by daniel(uid=0)
7 Aug 19 18:48:45 localhost sudo: pam_unix(sudo:session): session
  closed for user root
8 Aug 19 18:48:45 localhost sudo: daniel : TTY=pts/0 ; PWD=/home/
  daniel ; USER=root ; COMMAND=/usr/bin/apt-get -V dist-upgrade
9 Aug 19 18:48:45 localhost sudo: pam_unix(sudo:session): session
  opened for user root by daniel(uid=0)
10 Aug 19 18:48:46 localhost sudo: pam_unix(sudo:session): session
  closed for user root
11 ...

```

Listing 2.3: Auszug der Beispiel-Logdatei (auth.log)

Die Ergebnisse des Tests zeigen, dass große Unterschiede bei der Dauer der Kompression bestehen. `bzip2` benötigt gut 75 mal länger als `gzip`. Die zur Dekompression benötigte Zeit ist bei allen Verfahren ähnlich. Die mit Abstand beste Kompressionsrate wird von `xz` erzielt, das den Speicherverbrauch der Beispieldatei von 1 GB auf 154 KB senkt. Dies entspricht ungefähr einem Kompressionsfaktor von 6.500, während die anderen Verfahren einen deutlich niedrigeren Faktor aufweisen.

Tabelle 2.3: Testergebnisse des Kompressionstests

Methode	gzip	bzip2	xz
Dauer Kompression	15,69 s	1.177,96 s	329,76 s
Dauer Dekompression	24,90 s	37,96 s	19,21 s
Größe nach Kompression	6,0 MB	1,2 MB	0,154 MB
Kompressionsfaktor	167	833	~ 6.500
Eingesparter Speicherplatz	99,4%	99,88%	99,9846%

2.5 Schwierigkeiten der typischen Methoden

In diesem Abschnitt werden einige allgemeine Schwierigkeiten beschrieben, die bei den typischen Logging-Verfahren auftreten können. Das darauf folgende Kapitel geht darauf ein, wie Splunk Enterprise mit den erwähnten Schwierigkeiten umgeht.

2.5.1 Abweichende Syntax

Für jegliche automatisierte Verarbeitung von Logdateien ist es wichtig, dass die enthaltenen Daten einen ähnlichen Aufbau besitzen. Andernfalls kann ein verarbeitendes Programm keine korrekte Interpretation der pro Zeile enthaltenen Daten mehr erhalten.

So muss beispielsweise das Format der Markierung, wann ein Ereignis eingetreten ist beziehungsweise in die Logdatei geschrieben wurde (Zeitstempel) gleichartig sein und darf nicht zwischen einzelnen Ereigniszeilen verschieden formatiert sein. Dies kann aber der Fall sein, wenn mehrere Anwendungen in die gleiche Logdatei schreiben und dabei unterschiedliche Datumsformate verwenden.

Das folgende Listing enthält verschiedene Datumsformate, die eine korrekte Interpretation des Eintragsdatums durch ein zu simples Auswertungsprogramm verhindern würden.

```
1 ISO 8601                2014-08-24T18:25:57+02:00
2 Diverse/Linux         Aug 08 18:25:57
3 MySQL-DATETIME        2014-08-24 18:25:57
4 Apache (NCSA Common)  [24/Aug/2014:18:25:57 +0200]
5 ISC BIND               24-Aug-2014 18:25:57.107
```

Listing 2.4: Beispiele verschiedener Zeitstempel-Formate

Bereits die unterschiedlichen Darstellungsformen von Tag und Monat machen deutlich, dass eine textuelle Suche über mehrere Formate hinweg nicht ohne Weiteres möglich ist.

2.5.2 Wachstum von Logdateien

Sofern man keinen Anhaltspunkt hat, in welchem Zeitraum gesuchte Ereignisse aufgezeichnet worden sind und welche (rotierten) Logdateien diesem entsprechen, muss der komplette Datenbestand durchsucht werden. Der Aufwand hierfür steigt proportional zum erzeugten Logvolumen aller in Frage kommenden Systeme. Die Suche auf mehreren Systemen und beispielsweise das Aussortieren von Falsch-Positiven Ereignissen im selben Zeitraum kann unverhältnismäßig viel Zeit in Anspruch nehmen.

In einem Szenario mit verteilten Systemen gleicher Aufgabe kann ein einzelnes Ereignis auch auf mehreren Systemen protokolliert werden. So kann beispielsweise der Transfer einer E-Mail über mehrere E-Mailserver hinweg stattfinden. Um zum Beispiel die Ursache von Verzögerungen aufzuspüren, müssten die Logdateien auf allen Systemen händisch analysiert werden. Anschließend müsste das Gesamt ereignis aus den Einzelereignissen rekonstruiert werden.

Zusätzlich kann ein hohes Logvolumen bei größenabhängiger Rotation dazu führen, dass Logdaten zu schnell von den generierenden Systemen gelöscht werden und somit bereits nach kurzer Zeit nicht mehr zur Verfügung stehen.

2.5.3 Teure Negativsuche

Eine Suche nach nicht vorhandenen Daten stellt bei einer linearen Textsuche immer den aufwändigsten Fall dar. Denn um sicherstellen zu können, dass sich die gesuchten Daten auch nicht am Ende des zu durchsuchenden Bereichs befinden, muss der komplette Datenbestand durchsucht werden.

2.5.4 Verschiedenartige Werkzeuge

Die grundlegenden Werkzeuge, die zur Auswertung von Textdateien unter Linux eingesetzt werden können, bieten bereits umfassende Möglichkeiten. Allerdings sind sie in ihrer Bedienung oft auch sehr unterschiedlich. Standardwerkzeuge wie `grep` sind vergleichsweise leicht zu bedienen. Die Syntax von `awk` oder `find` hingegen nimmt schnell kompliziertere Ausmaße an. Für Werkzeuge wie `sed` sind fundierte Kenntnisse von regulären Ausdrücken erforderlich. Durch die uneinheitliche Bedienung wird das schnelle Umsetzen einer konkreten Aufgabe erschwert.

2.5.5 Zugriff auf Log-Journals

Einige Linux-Distributionen verwenden kein textbasiertes Logging sondern binäre, datenbankähnliche Speicherformen („Journals“). Die Distributionen *Fedora*¹² oder *Arch Linux*¹³ beispielsweise verwenden die Dienstesammlung `systemd` zur Systemverwaltung. Hier erfolgt der Zugriff auf systemnahe Logdaten ausschließlich durch eine eigene Softwarekomponente namens `journalctl`.

Zwar bietet diese Komponente auch Methoden zum Filtern und Durchsuchen der Datenbestände an, allerdings können diese nicht ohne Weiteres von anderen Anwendungen (wie beispielsweise Splunk) genutzt werden. [5] Die Abweichung vom Konzept des direkten Dateizugriffs geht also stark zu Lasten einer automatisierten Datenanalyse.

In einem Verbund aus mehreren Linux-Systemen unterschiedlicher Distributionen müssten folglich verschiedene Werkzeuge und Vorgehensweisen angewendet werden, um das selbe Ziel zu erreichen. Unabhängig davon, ob eine datei- oder journalbasierte Variante sich als vorteilhafter für ein konkretes Szenario darstellen mag, wird eine automatisierte Auswertung über mehrere Systeme hinweg schwieriger.

¹²<http://fedoraproject.org>

¹³<http://archlinux.org>

2.5.6 Nicht-unixoide Betriebssysteme

Andere Betriebssysteme, die keine unter Unix/Linux etablierten Verfahren zum Logging benutzen sind zunächst inkompatibel mit Techniken wie zentralisiertem syslog-Betrieb. In einem großen Netzwerk, das auf Diversität von Betriebssystemen setzt, zeigt sich hier der klare Nachteil einer fehlenden homogenen Logging-Infrastruktur.

Für Microsoft Windows beispielsweise existieren bis heute keine offiziellen syslog-Anbindungen. Man kann die Funktionalität nur durch zusätzliche Software von Drittanbietern nachrüsten.

Kapitel 3

Verbesserungspotential durch Splunk Enterprise

In diesem Kapitel soll erörtert werden, welche Vorteile Splunk gegenüber den klassischen Techniken bieten kann. Dabei wird Bezug auf die zuvor genannten Schwierigkeiten klassischer Systeme genommen.

3.1 Gleichartigkeit der Daten

Eine zentrale Fähigkeit von Splunk ist automatische Feldextraktion. Dahinter verbirgt sich eine automatische Erkennung von syntaktischen und logischen Abschnitten innerhalb von Logzeilen, genannt „Felder“. [6]

Zur Verdeutlichung folgt eine Logzeile eines Webservers im Combined Logfile Format.

```
1 192.109.234.216 - - [21/Aug/2014:01:33:24 +0200] "GET /index.html?
  kurs=123&matrikelnummer=123456 HTTP/1.1" 200 1468 "http://
  frankfurt-university.de/" "Mozilla/5.0 (X11; Ubuntu; Linux x86;
  rv:31.0) Gecko/20100101 Firefox/31.0"
```

Listing 3.1: Logbeispiel: Combined Logfile Format

Die Logzeile aus Listing 3.1 würde auf mehreren Ebenen von Splunk verarbeitet werden. Zum einen ist der Anwendung der Aufbau des Combined Logfile Format von vornherein bekannt. So würde jeder einzelne syntaktische Teil der Zeile (Erklärungen siehe Abschnitt 2.1) als Feld identifiziert werden. Auch die automatische Interpretation des Zeitstempels kann Splunk standardmäßig durchführen (siehe Abschnitt 5.1).

Zusätzlich zu den fest definierten Standardfeldern des CLF würden aus der Beispielzeile die Felder `kurs` und `matrikelnummer` als Felder erkannt werden. Dies geschieht, weil Splunk Zeichenketten links- und rechtsseitig von Gleichheitszeichen (unabhängig vom Log-Format) als Schlüssel/Wert-Paare interpretiert. [6]

Neben dem Wissen, wie das CLF zu interpretieren ist, sind Splunk auch andere verbreitete Logformate bekannt. Sollten für die Logdaten einer bestimmten Anwendung keine vordefinierten Extraktionsvorschriften existieren, können solche selbst angelegt oder nachinstalliert werden. [7]

3.2 Effizientes Durchsuchen

Durch das Aufteilen der rohen Textdaten in Felder und das anschließende Ablegen in bestimmte Indizes entsteht eine gleichartige Datenbasis innerhalb der Splunk-Infrastruktur. Such- und Auswertungsoperationen müssen nun nicht mehr auf Text ausgeführt werden sondern können anhand der Feldarten durchgeführt werden. Auf diese Weise werden auch Sortier- und Filteroperationen möglich, die auf textuellen Daten umständlich wären.

Negativsuchen stellen sich in Splunk effizienter dar als bei linearem Suchen auf Text. Zum Beispiel durch den Einsatz von sogenannten „Bloomfiltern“¹⁴ kann Splunk sehr schnell feststellen, dass der Datenbestand keine passenden Daten enthält. [8] Ein Bloomfilter basiert auf Hashtabellen und kann einen Negativfall daher mit konstantem Aufwand feststellen¹⁵. Die eigentliche Suchoperation muss in diesem Fall gar nicht mehr ausgeführt werden.

¹⁴<http://billmill.org/bloomfilter-tutorial>

¹⁵<http://www.sparknotes.com/cs/searching/hashtables/summary.html>

3.3 Zentrale Speicherung

Da der gesamte Logdatenbestand in Splunk-Indizes geschrieben wird, entfällt die Schwierigkeit, zusammengehörige Daten auf verschiedenen Systemen verwalten zu müssen. Die Index-tragenden Systeme können dabei auch geographisch verteilt sein. Sie können unabhängig von Ort und Anzahl als logische Einheit verschaltet werden und sind somit als zentralisiert zu betrachten.

Im Gegensatz zu einer zentralisierten syslog-Infrastruktur, in der Netzwerkprobleme zu Datenverlust führen können, wird die Zuverlässigkeit eines Splunk-Aufbaus sichergestellt. Dies wird in den Kapiteln 4 und 9 erklärt.

3.4 Langfristige Speicherung

Durch das Führen von zentralisierten Indizes werden Logdaten über lange Zeiträume vorgehalten. Eine standardmäßige logrotate-Konfiguration sieht vor, Logdateien bereits nach einigen Wochen vollständig vom System zu entfernen. Auch Splunk entfernt alte Daten schließlich aus den Indizes (siehe Abschnitt 5.3), jedoch passiert dies erst viel später. So kann der Speicherplatz auf den Quellsystemen früher freigegeben werden während die Logdaten weiterhin verfügbar sind.

3.5 Analyse und Aufbereitung

Die indizierten Logdaten aller beteiligten Systeme können durch die Web-Oberfläche von Splunk durchsucht werden. Dies geschieht in der Regel schneller als eine Suche auf den rohen, nicht indizierten Textdaten. Die ermittelten Suchergebnisse können hier bereits sortiert oder verfeinert werden.

Ein konkreter Suchvorgang kann abgespeichert werden, sodass er für einen erneuten Aufruf zur Verfügung steht ohne die komplette Suche erneut ausführen zu müssen. [9]

Splunk bietet umfangreiche Möglichkeiten zur visuellen Darstellung von Suchergebnissen. Diese Aufbereitungen können spontan erzeugt oder für erneute Verwendung auf anderen Daten als „Reports“ gespeichert werden. Für eine kontinuierliche Darstellung von Ereignissen können mehrere Reports auf visuellen „Dashboards“ angeordnet werden. Solche Flächendarstellungen können genutzt werden, um mehrere thematische Gruppen von Reports vorzuhalten. Diese können dann beispielsweise auf Bildschirmen in Büroräumen dargestellt werden (siehe Beispiele im Abschnitt 8.2). [10]

3.6 Search Processing Language (SPL)

Zur Formulierung der Suchanfragen und Verarbeitungsanweisungen nutzt Splunk eine eigene Befehlssprache namens „Search Processing Language“ (SPL). Sie ist syntaktisch einfach gehalten und umfangreich dokumentiert. [5][9][11][12]

Sie enthält viele Funktionsanweisungen zur Weiterverarbeitung ermittelter Suchergebnisse. Durch Verkettung von einzelnen Anweisungen können Suchen beispielsweise von der selben Befehlszeile aus gefiltert, gruppiert und an eine Visualisierungsfunktion weitergereicht werden.

Die Nutzung der SPL erspart weniger technik-orientierten Anwendern die Notwendigkeit, sich mit Linux-Kommandozeilen und -Werkzeugen auseinandersetzen zu müssen.

3.7 Systemunabhängigkeit

Splunk stellt alle relevanten Software-Pakete auch für Microsoft Windows und andere Betriebssysteme nativ zur Verfügung, leistet Support für diese und liefert Aktualisierungen.

Kapitel 4

Komponenten von Splunk

Auf jedem System, das eine der im Folgenden näher betrachteten Splunk-Funktionalitäten ausführen soll, wird das gleiche allgemeine Softwarepaket¹⁶ installiert. Die Festlegung, welche der Funktionen letztendlich vom System übernommen wird, findet jeweils durch Konfiguration der Instanz statt. [13]

Den einzigen Sonderfall stellt der „Universal Forwarder“ dar, der als separates Installationspaket¹⁷ verfügbar ist und die generierten Logdaten eines Systems an Splunk übermittelt. Er ist als möglichst leichtgewichtige Anwendung für die Systeme gedacht, auf denen ansonsten keine weitere Splunk-Funktionalität benötigt wird. [14]

Die Installation des allgemeinen Softwarepakets auf einem einzelnen System stellt bereits eine vollständige Splunk-Umgebung bereit. Diese bietet die Möglichkeit, Logdaten von anderen Systemen entgegenzunehmen, zu indizieren und für Suchanfragen bereit zu halten. Die Bedienung erfolgt über eine Web-Oberfläche, welche ebenfalls in der Basis-Installation enthalten ist.

Falls solch ein zentralisierter Einzelaufbau den Ansprüchen aus Performanz- oder infrastrukturellen Gründen nicht gerecht wird, können die einzelnen Splunk-Komponenten auf separate Systeme verteilt werden. Ein Szenario mit dedizierten Systemen für einzelne Splunk-Komponenten wird „Cluster“ genannt. Innerhalb eines Clusters lassen sich Splunk-Komponenten wie Indexer oder Search Heads zu „Pools“ zusammenfassen. [8][15]

¹⁶<http://www.splunk.com/download>

¹⁷<http://www.splunk.com/download/universalforwarder>

4.1 Indexer

Der Indexer ist der wichtigste Teil einer Splunk-Infrastruktur und besteht aus mindestens zwei aktivierten Teilkomponenten, einem „Receiver“ und einem „Search Peer“.

Die Receiver-Komponente nimmt über das Netzwerk eingehende oder lokal erfasste Logdaten auf, behandelt sie in der sogenannten „Parsing-Pipeline“ und indiziert sie in einer „Indexing Pipeline“ (siehe Kapitel 5). Auf Basis der nun indizierten Informationen können später Antworten auf Suchanfragen erzeugt werden.

Die Search-Peer-Komponente eines Indexers verbindet den Datenbestand des Indexers mit der Nutzeroberfläche, welche sich auf dem Indexer selbst oder auf einem dedizierten System befinden kann. [8]

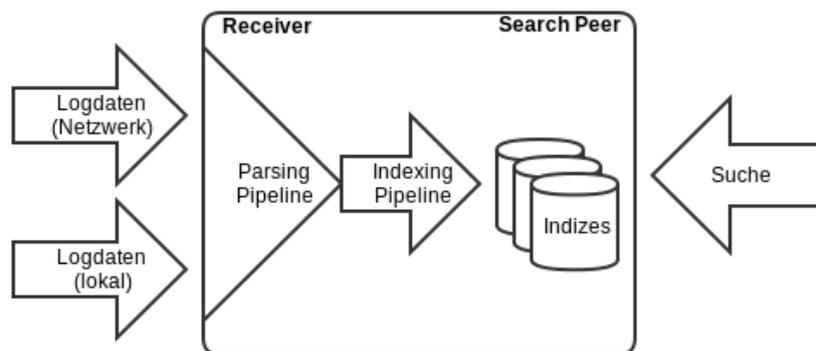


Abbildung 4.1: Schematische Darstellung des Splunk-Indexers

Ein Indexer stellt somit den Kern von jedem Splunk-Aufbau dar. Er übernimmt alle Funktionen, sofern keine Auslagerung auf dedizierte Systeme durchgeführt wird. Eine ausgelagerte Funktionalität kann oder muss auf dem Indexer deaktiviert werden. Beispielsweise kann ein Cluster über einen dedizierten Search Head verfügen, um die Nutzeroberfläche bereitzustellen während die Nutzeroberfläche des Indexer zusätzlich verfügbar bleibt. Die Funktionalität der Lizenzverwaltung hingegen kann nur von einem System übernommen werden. Sie würde auf dem Indexer deaktiviert, sofern der Cluster gegen einen dedizierten Lizenz-Server konfiguriert wird.

Als Systemanforderungen für Indexer werden die folgenden Werte genannt. [11][13]

Tabelle 4.1: Systemanforderungen

	minimal	empfohlen
CPU	1,4 GHz	2x6 Kerne 2+ GHz
RAM	1 GB	12 GB
HDD	5 GB + Indizes	+ RAID 0/10

Ein Indexer mit den empfohlenen Spezifikationen kann 20 MB an Logdaten pro Sekunde indizieren. Für die nicht-indizierenden Komponenten reichen jeweils ein 1,5 GHz-Prozessor und 1 GB RAM aus. [13]

4.2 Forwarder

Forwarder leiten gesammelte Logdaten an Indexer weiter. Es handelt sich dabei um ein auf TCP¹⁸ basierendes Protokoll namens „SplunkTCP“. [14] Es besteht kein Zwang, Forwarder zu nutzen, da Splunk beispielsweise auch über das syslog-Protokoll eingelieferte Datenströme verarbeiten kann. Forwarder bieten allerdings einige Vorteile gegenüber syslog.

Ein Forwarder kann dahingehend konfiguriert werden, Empfangsbestätigungen für übermittelte Logdaten abzuwarten. Erfolgt keine Bestätigung, wird weiterhin versucht, die betreffenden Daten an einen Indexer zu übermitteln. Bei der Übermittlung zwischen syslog-Instanzen ist dies nicht möglich und einmal übermittelte Daten, die zum Beispiel aufgrund von Netzwerkproblemen ihr Ziel nicht erreichen, sind verloren.

Forwarder verfügen außerdem über eigene Mechanismen zur Lastverteilung. Wird ein Forwarder gegen mehrere Indexer eines Clusters konfiguriert, teilt der Forwarder die ausgehende Datenmenge automatisch unter ihnen auf. Es besteht somit kein weiterer Bedarf an Lastverteilung durch zusätzliche Netzwerk-Hardware. [14]

¹⁸Transmission Control Protocol

4.2.1 Universal Forwarder

Der „Universal Forwarder“ ist die einfachste Komponente des Splunk-Frameworks. Seine einzige Aufgabe ist es, Logdaten des lokalen Systems entgegenzunehmen und in roher Form an einen Splunk-Indexer oder einen Pool von Splunk-Indexern weiterzuleiten. [14] Dieses Verhalten entspricht dem eines syslog-Dienstes, der Logdaten weiterleitet anstatt sie selbst zu verarbeiten. Hierbei profitiert er allerdings von den zuvor erwähnten Vorteilen der Empfangsbestätigungen und Lastverteilung.

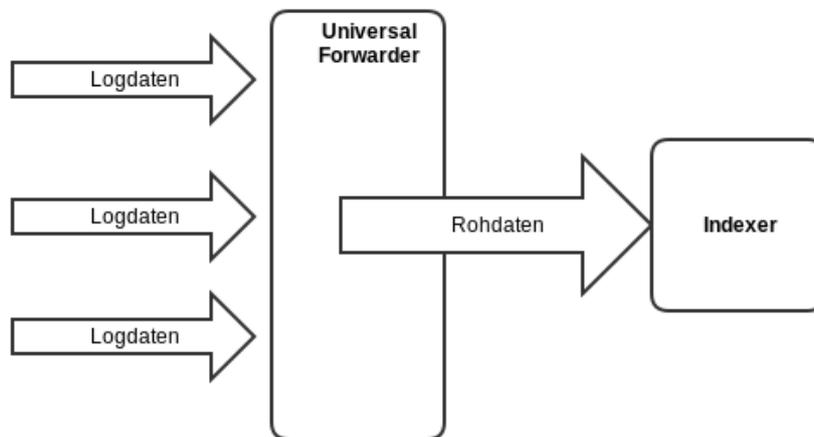


Abbildung 4.2: Schematische Darstellung des Universal Forwarder

Der Universal Forwarder ist die einzige Splunk-Komponente, die als eigenständiges Installationspaket existiert. Durch seine vergleichsweise geringe Anwendungsgröße und Ressourcenbedarf ist er darauf ausgelegt, auf vielen Systemen installiert zu werden, ohne jeweils eine volle Installation des allgemeinen Splunk-Paketes vornehmen zu müssen. [14]

4.2.2 Heavy Forwarder

Der „Heavy Forwarder“ hat wie der „Universal Forwarder“ ebenfalls die Hauptaufgabe, lokal erhaltene Logdaten an Splunk-Indexer weiterzuleiten. Es sind allerdings zwei Unterschiede zum Universal Forwarder anzumerken: Erstens besitzt der Heavy Forwarder kein eigenes Installationspaket sondern wird aus einer allgemeinen Splunk-Installation heraus konfiguriert. Zweitens leitet der Heavy Forwarder keine Rohdaten an die Indexer weiter, sondern führt bereits das Parsing der Daten aus (siehe Kapitel 5). Somit

wird der Gesamtaufwand der Indizierung zwischen Heavy Forwarder und Indexer aufgeteilt. Der Indexer muss anschließend nur noch die eigentliche Indizierung der bereits geparsten Daten vornehmen. [14]

Der Heavy Forwarder kann die vorbehandelten und weitergeleiteten Daten zusätzlich in eigene Indizes aufnehmen.

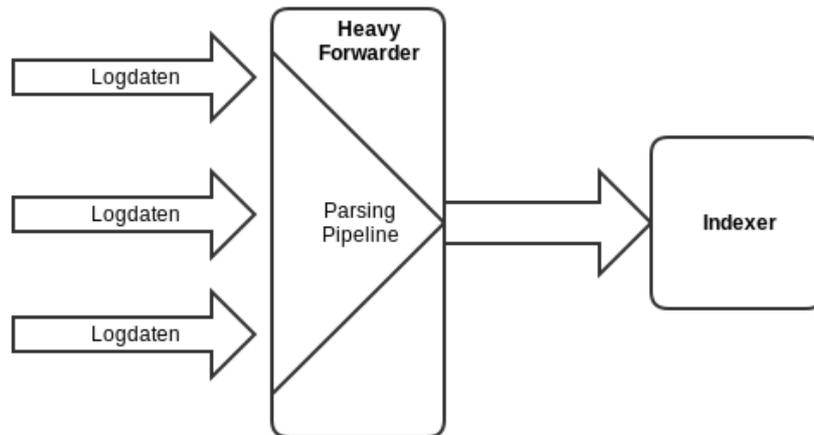


Abbildung 4.3: Schematische Darstellung des Heavy Forwarder

In einem klassischen Cluster-Aufbau ist die Verwendung von leichtgewichtigen Universal Forwardern den Heavy Forwardern aus Gründen von geringerem Ressourcenbedarf und des Administrationsaufwand vorzuziehen.

4.3 Search Heads

Der Search Head stellt die Benutzeroberfläche in einem Splunk-Cluster bereit. Er koordiniert die eingegebenen Suchanfragen und vermittelt sie an die Indexer des Clusters. Er nimmt auch die Antworten der Indexer entgegen und konsolidiert diese für den Benutzer. Suchen können im Search Head gespeichert werden, um zu späteren Zeitpunkten auf die Ergebnisse zurückzugreifen. [15]

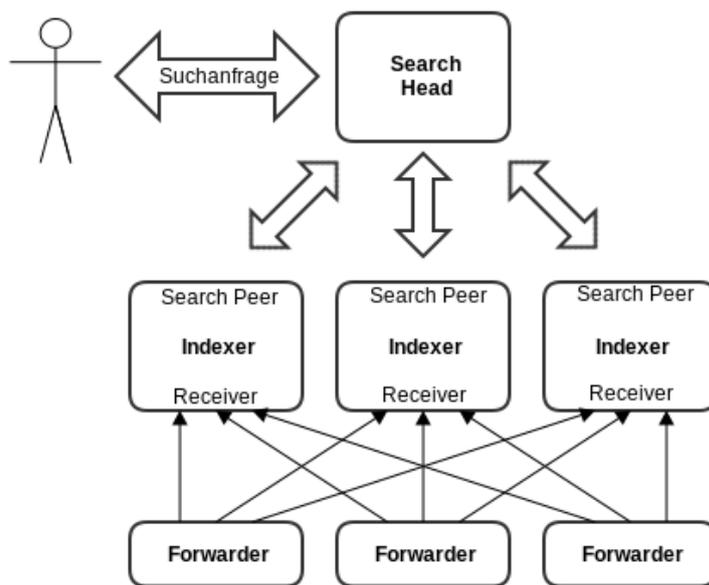


Abbildung 4.4: Schematische Darstellung eines Clusters mit Search Head

Der Search Head regelt auch die Autorisierung von Benutzern in Bezug auf das Abfragen bestimmter Indizes. Die Informationen über den Autorisierungsstatus eines Benutzers werden zusammen mit den Suchanfrage an die Indexer übermittelt. Auch zusätzlich installierte Konfigurationspakete, die Einfluss auf die Suchanfrage haben, werden übermittelt. Das Paket, welches diese Informationen enthält heißt „Knowledge Bundle“ [15]. Sollten Search Head(s) und Indexer geographisch verteilt sein, ist bei der Planung des Aufbaus zu beachten, dass auch das Knowledge Bundle transferiert werden muss und aufgrund seiner Größe eventuellen Einfluss auf die Netzwerklast haben könnte.

Ein Cluster muss über mindestens einen Search Head verfügen. Zwar können die Nutzeroberflächen auf den Indexern gleichzeitig aktiviert bleiben, allerdings ist nicht garantiert, dass zu einem bestimmten Zeitpunkt jeder Indexer den kompletten Datenbestand kennt. Somit würde eine auf einem Indexer ausgeführte Suche möglicherweise nur eine Teilmenge der gesuchten Daten zurück liefern. [8]

In einem fortgeschrittenen Aufbau können mehrere „Search Heads“ zu einem Pool verbunden verbunden werden um hohe Anfragelasten auf mehrere Systeme zu verteilen. Im Gegensatz zu mehreren einzelnen Search Heads würden in einem Pool die Informationen, aus denen das Knowledge Bundle erzeugt wird, untereinander repliziert werden. [15]

Es wird empfohlen, den Search Head eines Clusters auch als lizenzverwaltende Komponente zu definieren (siehe Kapitel 7.6). [16]

4.4 Master Nodes

Jeder Cluster wird von einem Master Node koordiniert. Er steuert die Indexreplikation zwischen den Indexern und teilt einem anfragenden Search Head mit, auf welchen Indexern die gesuchten Daten vorliegen. Beim Ausfall eines Indexers leitet der Master Node die notwendigen Aktionen ein, um die nicht mehr zugreifbaren Datenbestände auf anderen Indexern verfügbar zu machen. Danach vermittelt er Suchanfragen von Search Heads um das fehlerhafte System herum. [8]

Kapitel 5

Splunk-interne Strukturierung der Daten

Der Weg von jeglichen Logdaten hinein in einen Splunk-Index führt zunächst durch eine Parsing Pipeline und danach durch eine Indexing Pipeline. Letztere befindet sich immer auf dem System, welches auch den Index selbst beherbergt. Im Falle von Cluster-Betrieb durchlaufen die Daten die Indexing-Pipeline des Peer Nodes (Indexer eines Clusters), durch den die Daten den Cluster betreten. Die Parsing Pipeline befindet sich in der Regel auch dort, sofern die eingehenden Daten nicht von einem Heavy Forwarder gesendet und dort bereits geparkt wurden (siehe Abschnitt 4.2.2).

Die Indizes, in welche die Daten letztendlich eingefügt werden, bestehen jeweils aus mehreren Binärdateien, die „Buckets“ genannt werden. Durch die Vorhaltung in solchen Buckets wird auch ein Rotations- und Archivierungskonzept umgesetzt (siehe Kapitel 2.3).

Der Prozesse innerhalb der Pipelines und die Verwaltung der Buckets sind komplex und werden im Folgenden nur zur Verdeutlichung beschrieben. Sie sind in den Splunk-Dokumentationen ausführlich dokumentiert. [7][8]

5.1 Parsing

In der Parsing Pipeline des Indexers (oder des Heavy Forwarders) werden die Daten den folgenden Operationen unterzogen:

Feststellen des Zeichensatzes: Splunk arbeitet intern mit dem UTF-8-Zeichensatz. Treffen Daten unter anderen Zeichensätzen ein, werden diese zur weiteren Verarbeitung in UTF-8 umgewandelt.

Aufteilen in Ereignisse: Ein Datensatz oder Datenstrom muss nicht zwangsläufig in einzelne Ereignisse getrennt eintreffen. Daher ist es für alle weitere Verarbeitungsschritte zunächst nötig, einzelne Ereignisse zu identifizieren und mit entsprechenden Endmarkierungen zu versehen. Ein Ereignis kann, beispielsweise bei zeilenbasiertem Logging, auch aus mehreren Zeilen bestehen, weshalb die Erkennung aufwändig sein kann. Für einige mehrzeilige Ereignisse kennt Splunk die Regeln, nach denen Ereignisse voneinander zu trennen sind, für unbekannte Formate müssen eigene Muster definiert werden.

Standardfelder extrahieren: Unabhängig von der Art der eingehenden Daten können aus Ereignissen immer Felder wie `host`, `splunk_server`, `source`, `sourcetype` und viele weitere extrahiert werden. Bei ihnen handelt es sich sozusagen um „Metadaten“, die nicht innerhalb der eigentlichen Ereignisse zu finden sind sondern vom sendenden Splunk-System angefügt wurden. `host` enthält dabei den Namen des sendenden Systems, `splunk_server` den Namen des indizierenden Systems, `source` die konkrete Quelle (beispielsweise der Name der ursprünglichen Logdatei) und `sourcetype` beschreibt die Art der konkreten Quelle (z.B. Textdatei, syslog-Datenstrom, Daten im Combined Logfile Format oder sonstige).

Zeitstempel erkennen: Der Zeitstempel des Ereignisses, der auf verschiedene Weisen strukturiert sein kann (siehe Abschnitt 2.5.1), wird identifiziert und im ISO8601-Format vereinheitlicht. Um den Zeitstempel eines Ereignisses zu erkennen, führt Splunk verschiedene Aktionen aus.

Zunächst wird versucht, ein zeitstempelähnliches Muster innerhalb der Rohdaten zu finden. Scheitert dies, werden folgende Versuche der Reihe nach unternommen, bis einer erfolgreich ist:

- anhand des `sourcetype` des Ereignisses prüfen, ob ein Zeitstempel an anderer Stelle außerhalb der Rohdaten zu finden ist (z.B. in den zuvor extrahierten „Metadaten“ oder im Netzwerkprotokoll, über das der Datensatz kam), bei Erfolg verwenden
- den Zeitstempel des vorherigen Ereignisses mit selbem `sourcetype` verwenden (sofern ein vorheriges Ereignis existiert)
- das `source`-Feld des Datensatzes (enthält in der Regel einen Dateinamen) auf Zeitstempelmuster prüfen und bei Erfolg verwenden
- das Datum der letzten Dateibearbeitung verwenden (nur im Falle von Logdateien als Datenquelle möglich)
- dem Ereignis die aktuelle Zeit zuweisen

Der Zeitstempel wird in Suchen als Standardfeld `_time` angegeben.

Anonymisierung: Zum Unkenntlichmachen von kritischen Daten wie Kreditkartennummern, IP-Adressen oder Passwörtern können Regeln definiert werden, nach denen solche Daten maskiert werden und somit nicht in die Indizes gelangen.

5.2 Indexierung

Die gepushten Ereignisse erreichen nun die Indexing-Pipeline. Hier werden die folgenden Aktionen durchgeführt:

Segmentierung der Ereignisse: Segmente sind die durchsuchbaren Teile eines Ereignisses, vergleichbar mit den syntaktischen Feldern einer Textlogzeile. Unterschieden werden dabei „innere“, „äußere“ und „volle“ Segmentierung. In Bezug auf eine Uhrzeit wäre beispielsweise `12:23:34 AM` das äußere Segment, die inneren Segmente wären `12`, `23`, `34` und `AM`. Die

volle Segmentierung würde sowohl innere als auch äußere Segmente in die Indizes schreiben. Splunk kennt einige Regeln zur Segmentierungen von typischen Daten (z.B. Uhrzeiten, IP-Adressen) von vornherein, andere können konfiguriert werden.

Verteilen der Segmente auf Buckets: Die Bedeutung von Buckets wird im nächsten Abschnitt genauer beschrieben.

Finalisierung Die Datenstrukturen der Indizes (Buckets) werden auf die Festplatten geschrieben und nachträglich („post-indexing“) komprimiert.

5.3 Buckets

Ein Index besteht aus mehreren Buckets. Jeder Bucket besteht aus zwei Dateiformen: Komprimierten Rohdaten und jeweils dazugehörigen Indexdateien. Anhand der Indexdateien lässt sich ermitteln, an welcher Stelle der komprimierten Rohdaten die von einer Suche angeforderten Abschnitte zu finden sind. Durch das „Nachschlagen“ in den Indexdateien geht das Auffinden von Abschnitten in den Rohdaten erheblich schneller als wenn die Rohdaten dafür komplett durchsucht werden müssten. Durch den Einsatz von weiteren Indizierungstechniken wie Bloomfiltern kann das Nichtvorhandensein von relevanten Abschnitten mit konstantem Aufwand ermittelt werden, ohne dass überhaupt eine Suche auf den Rohdaten stattfinden muss.

Da Buckets auf Dateiebene implementiert sind, kann durch Rotationsverfahren (siehe Kapitel 2.3) auch hier ein effizienter Kompromiss zwischen benötigtem Speicherplatz und Relevanz der vorgehaltenen Daten angestrebt werden. Im Gegensatz zur einfachen, zeit- oder größenabhängigen Rotation von einzelnen Logdateien, geht Splunk hierbei nach besonderen Kriterien und Techniken vor.

Jeder Bucket befindet sich in einem von fünf möglichen Zuständen:

Hot: „Heiße“ Buckets enthalten die aktuellsten Daten. Das Betriebssystem hält Dateizeiger offen, neu indizierte Daten werden hier eingeschrieben. Heiße Buckets sind auf **Indizierungsgeschwindigkeit** optimiert.

Warm: „Warme“ Buckets sind erstmals rotierte, ehemals heiße Buckets. Der Unterschied besteht darin, dass keine Dateizeiger des Betriebssystems mehr auf den Bucket zeigen. Da keine Schreibzugriffe mehr stattfinden, können auf warme Buckets splunk-interne Optimierungen der **Suchgeschwindigkeit** angewendet werden.

Cold: „Kalte“ Buckets sind ehemals warme, die eine weitere Rotation durchlaufen haben. Auch kalte Buckets sind auf **Suchgeschwindigkeit** optimiert. Im Gegensatz zu warmen Buckets, die sich weiterhin die Verzeichnisstrukturen mit heißen Buckets teilen, werden sie aber in gesonderte Bereiche des Dateisystems verschoben.

Frozen: „Eingefroren“ beschreibt den Zustand eines Buckets nach einer weiteren Rotation. Der Begriff „Zustand“ kann hier irreführend sein, da beim „einfrieren“ eines Buckets eine spezielle Aktion ausgeführt wird. Standardmäßig ist diese Aktion das Löschen des Buckets.

Alternativ kann Splunk aber veranlasst werden, Buckets beim Einfrieren nicht zu löschen sondern an einen Ort außerhalb der Splunk-Verantwortlichkeit zu verschieben. Ab hier passt das Sinnbild eines eingefrorenen Datenbestandes wieder, denn solche Buckets können zu späteren Zeitpunkten noch einmal in Splunk geladen und durchsucht werden. Erwartungsgemäß werden vormalig eingefrorene und wieder integrierte Buckets als „aufgetaut“ bezeichnet.

Thawed: Vormalig „eingefrorene“ und nun wieder integrierte Buckets. Sie unterliegen keinen weiteren Rotation sondern verbleiben in diesem Zustand. Nach Abschluss der Arbeiten, wegen denen das Auftauen stattfand, können sie manuell gelöscht werden.

Tabelle 5.1: Kriterien für Bucketrotation

von	nach	Kriterien
hot	warm	Bucketgröße oder Alter des ältesten enthaltenen Datensatzes überschreitet Grenzwert
warm	cold	Anzahl gleichzeitig vorgehaltener warmer Buckets überschreitet Grenzwert
cold	frozen	alle Buckets des Systems addiert überschreiten maximale Größe
frozen	(gelöscht)	bei Rotation eines kalten Bucket
frozen	(archiviert)	bei Rotation eines kalten Bucket, sofern konfiguriert
(archiviert)	thawed	manuell
thawed	(gelöscht / archiviert)	manuell

Diese Tabelle verdeutlicht die, im Vergleich zu klassischem logrotate komplexeren Kriterien, wann Buckets in den Folgezustand rotiert werden. Anzumerken ist außerdem, dass bei einem Systemneustart alle Buckets rotiert werden.

Der Speicherbedarf der Indizes beträgt etwa (*Größe der Rohdaten * 0,5*). Dabei machen die komprimierten Logdaten etwa 10% dieses Speicherbedarfs aus, der Rest wird von den dazu gehörigen Indexdateien beansprucht. [13]

Kapitel 6

Typische Einsatzszenarien

Je nach Anforderungen kann eine Splunk-Infrastruktur aus einem einzelnen System bestehen und darüber hinaus schrittweise zu größeren verteilten Szenarien erweitert werden. Dieses Kapitel soll eine Übersicht über die verschiedenen logischen Szenarien und über ihre jeweiligen Vor- und Nachteile geben.

6.1 Einzelner Splunk-Server

Das einfachste Szenario zum Einsatz von Splunk ist die Installation der Software auf einem einzelnen System. Hierbei übernimmt das System alle Funktionen selbst, vom Empfang externer Logdaten, über die Indexierung bis hin zur Bereitstellung der Such- und Analysefunktionen. [13]

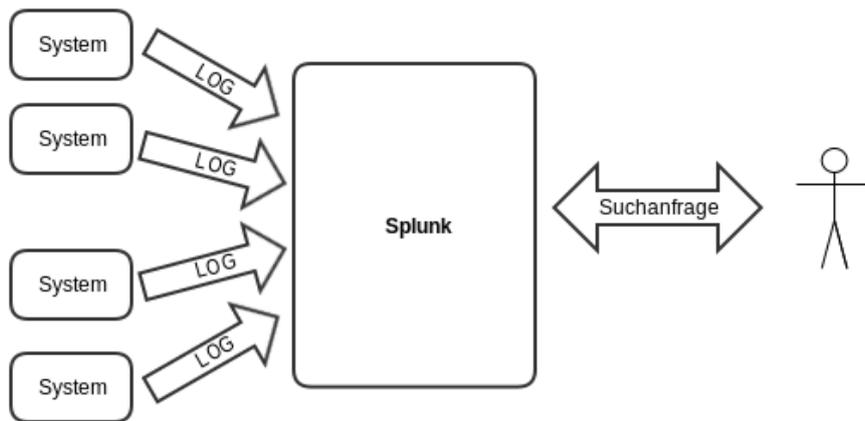


Abbildung 6.1: Einzelner Splunk-Server

Dieses Szenario ist aufgrund seiner geringen Komplexität vorteilhaft, da keine Komponenten auf andere Systeme verteilt werden müssen. Auch das Einspielen von Aktualisierungen und neuen Versionen bedarf keiner weiteren Koordination mit anderen Systemen.

Für diese Betriebsvariante muss betrachtet werden, ob die Performanz des Einzelsystems ausreicht, um alle eingehenden Logdaten indizieren und zusätzlich noch die Suchanfragen von Benutzern verarbeiten zu können. Als Richtwerte für den optimalen Betrieb eines Einzelsystems werden ein Indextvolumen von weniger als 50 GB pro Tag und weniger als 10 parallele Benutzer genannt. [5]

Das Einzelsystem wird hier allerdings zu einem kritischen Punkt der Infrastruktur. Ein Ausfall der Hardware des Systems oder netzwerkbedingte Erreichbarkeitsprobleme haben direkt den Ausfall sämtlicher Logging-Kapazitäten zur Folge. Es kommt zwar zu keinen Datenverlusten, da die Forwarder der Infrastruktur keine Empfangsbestätigungen vom ausgefallenen System mehr erhalten, allerdings stehen den Benutzern natürlich auch keine Splunk-Funktionalitäten mehr zur Verfügung.

Zur Absicherung gegen derartige Schwierigkeiten kann eine Splunk-Infrastruktur auf verschiedene Weisen verteilt werden.

6.2 Mehrere Indexer

In einem Aufbau mit mehreren separaten Indexern muss der Ausfall eines Systems nicht mehr den Totalausfall der Logging-Infrastruktur zur Folge haben. Werden gemäß der Variante mit einem einzelnen Splunk-System mehrere Indexer eingerichtet, so verfügt jeder von ihnen über eine Teilmenge der gesamten eingelieferten Logdaten.

Dies kann bewerkstelligt werden, indem die Quellsysteme der eingehenden Daten auf verschiedene Indexer hin konfiguriert werden. In diesem Fall wissen die einzelnen Indexer nicht, dass es weitere im Verbund gibt, da sie jeweils als Einzelsystem konfiguriert sind.

Ein Ausfall eines Indexers hat zwar keinen Totalausfall mehr zur Folge, allerdings besteht kein Zugriff auf die Datenbestände des jeweiligen Indexers mehr. Ein weiterer Nachteil dieser Variante ist, dass der Benutzer von vornherein wissen muss, auf welchem Indexer sich die für ihn relevanten Daten befinden.

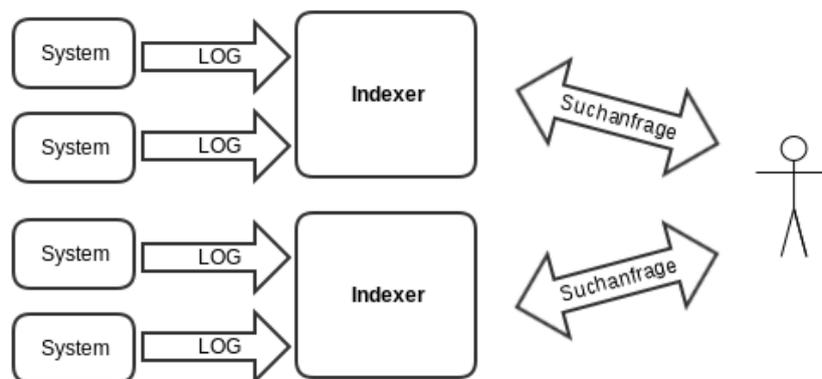


Abbildung 6.2: Mehrere separate Indexer

Um die disjunkte Verteilung von Logdaten auf mehrere Indexer zu vermeiden, können mehrere Indexer als Cluster zusammengefasst werden. Innerhalb eines solchen Clusters werden die gesammelten Daten unter den beteiligten Indexern repliziert, sodass jeder über den vollständigen Datenbestand verfügt. Ausfälle einzelner Indexer sind hierbei weniger kritisch. Sie können allerdings noch negativen Einfluss auf die Performanz der Infrastruktur haben, da die verbleibenden Indexer nun die Arbeitslast des ausgefallenen Systems mit übernehmen müssen. Pro Indexer sollte ein Datenvolumen von 100 GB pro Tag nicht überschritten werden, um ausreichende Performanz des Systems gewährleisten zu können [5].

Der Benutzer muss hier nicht wissen, auf welchem System die relevanten Daten gespeichert sind, da sich die Indexer untereinander replizieren. Es reicht aus, die Web-Oberfläche eines beliebigen Indexers aufzurufen um Zugriff auf den gesamten Datenbestand zu erhalten. Idealerweise erfolgt der Benutzerzugriff aber nicht direkt auf eine Indexer-Adresse sondern auf einen vorgeschalteten, netzwerkseitigen Lastverteiler (load balancer). Parallele Abfragen von mehreren Benutzern, die ansonsten vielleicht alle den gleichen Indexer angesteuert hätten, gehen stattdessen an den load balancer und erzeugen keine ungleiche Lastverteilung im Cluster.

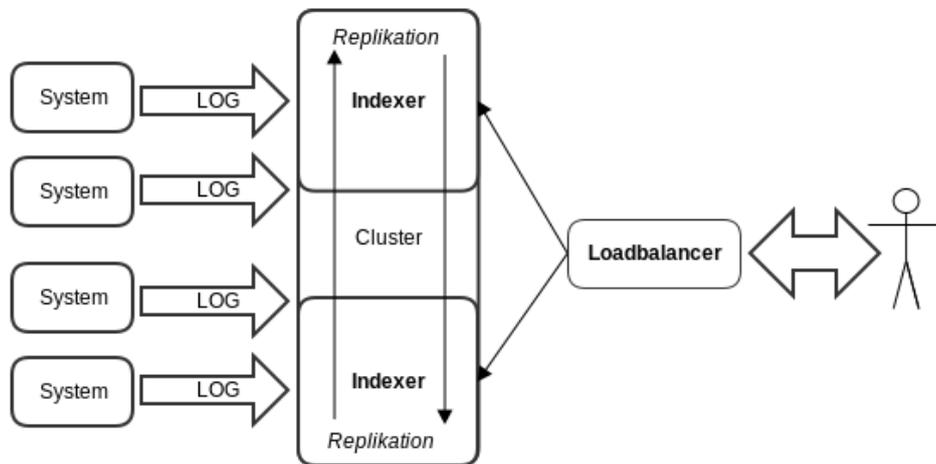


Abbildung 6.3: Cluster mit Lastverteilung

6.3 Indexer pro Standort mit einem zentralen Search Head

Sollen mehrere geographisch getrennte Standorte abgedeckt werden, so könnte auch dies mit dem zuvor beschriebenen Szenario erreicht werden. Die einzelnen Systeme eines entfernten Standorts könnten die Indexer des zentralen Standorts über das Internet erreichen. Diese Variante beinhaltet allerdings einige Nachteile. So muss berücksichtigt werden, dass der Transfer sämtlicher Daten kontinuierlichen Netzwerkverkehr zwischen den entfernten und dem zentralen Standort verursachen würde. Außerdem wären die entfernten Standorte auf eine fehlerfreie Verbindung zum zentralen Standort angewiesen. Zudem kann es dem Sicherheitskonzept des Netzwerks widersprechen, dass entfernte Standorte eine Verbindung in den zentralen Standort hinein aufbauen können.

Um diese Nachteile zu überwinden, kann an jedem Standort ein eigener Indexer aufgestellt werden. Dieser nimmt in gewohnter Weise die Logdaten des jeweiligen Standorts entgegen und verwaltet sie. Um Suchanfragen über die einzelnen Standorte hinweg durchführen zu können, wird am zentralen Standort ein zentraler Search Head eingerichtet. Dieser wird gegen die entfernten Indexer konfiguriert und leitet Suchanfragen zu ihnen weiter. [15]

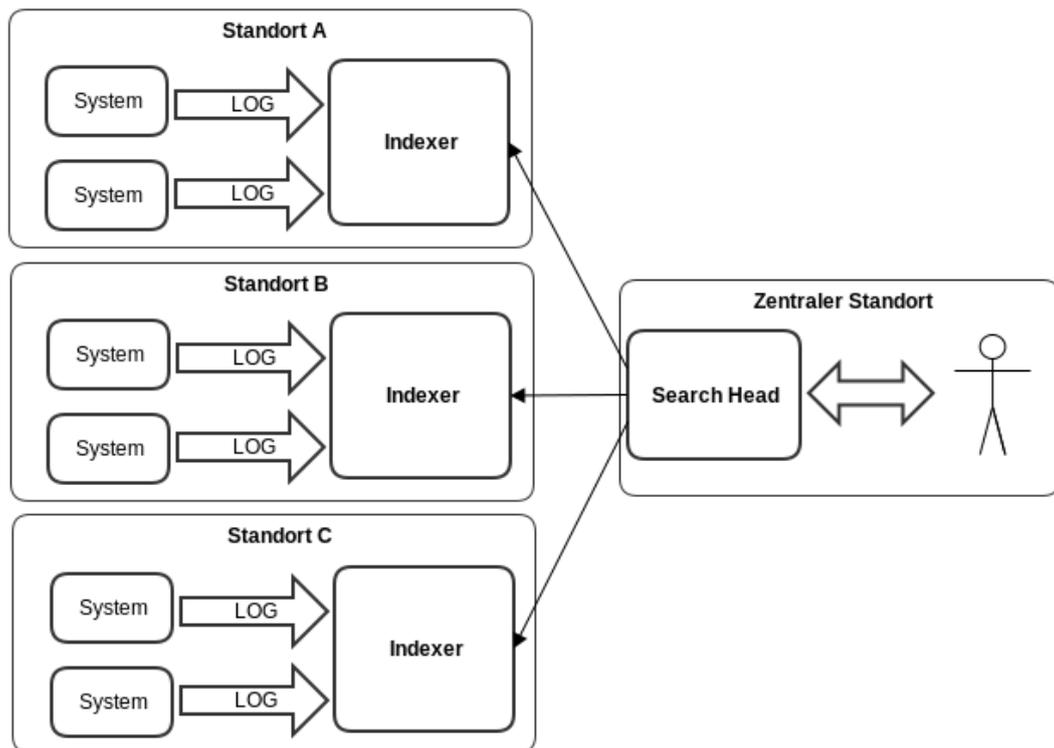


Abbildung 6.4: Splunk-Infrastruktur über mehrere Standorte

Fällt die Verbindung zu einem entfernten Standort aus, kann zwar vom zentralen Search Head aus nicht mehr auf die Datenbestände zugegriffen werden, allerdings gehen keine Daten verloren, da diese im Indexer des Standorts vorgehalten werden.

6.4 Search Head Pooling und Indexer-Cluster

Ähnlich wie bei einem Cluster von Indexern können auch mehrere Search Heads gruppiert werden. Solch ein Verbund heißt „Search Head Pool“ und dient in erster Linie zur Splunk-internen Lastverteilung der Suchanfragen. Zusätzlich dazu werden Benutzerprofile, gespeicherte Suchen und nachinstallierte Konfigurationspakete unter den Search Heads des Pools repliziert. [15]

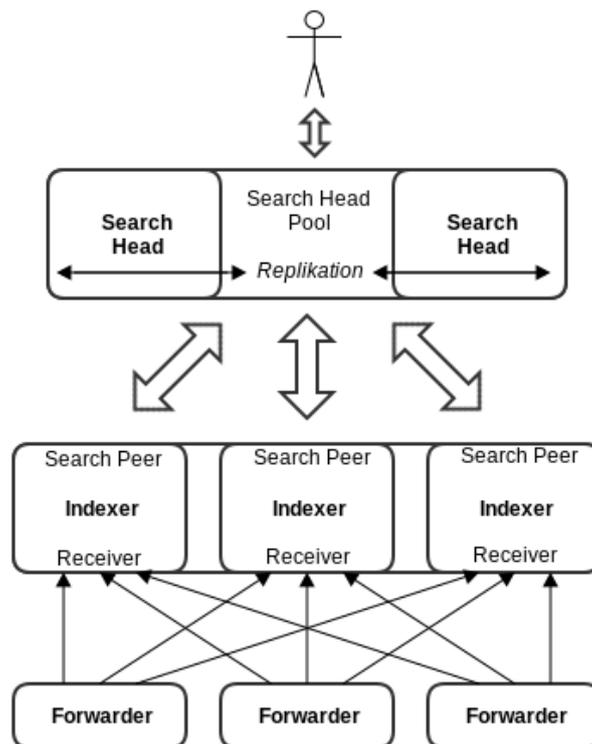


Abbildung 6.5: Schematische Darstellung eines Clusters mit Search Head Pool

Kapitel 7

Implementierung der Splunk-Infrastruktur

In diesem Kapitel folgt zunächst eine Beschreibung der bestehenden Logging-Verfahren am zentralen und an den verteilten Standorten. Danach wird das Zielbild der neuen Infrastruktur vorgestellt und schrittweise die Implementierung der einzelnen Komponenten gezeigt.

Zur Sammlung der gezeigten Konfigurationsabschnitte und Abbildungen wurden auf bereitgestellter Hardware im DENIC-Netzwerk zwei Splunk-Testcluster aufgebaut.

7.1 Bestehende Lösungen am zentralen Standort

Die Linux-Systeme des zentralen Standorts, von dem aus die Domain-Registrierungsdienste angeboten werden, sind klassisch konfiguriert (siehe Kapitel 2) und halten ihre generierten Logdaten lokal vor. Das unter Linux typische Zusammenspiel von syslog und logrotate wird eingesetzt. Es findet keine zentrale Konsolidierung der Logdaten über einzelne Systemgrenzen hinaus statt. Die einzige Ausnahme bilden hierbei Netzwerkgeräte wie Router, Switches und Firewalls, deren Logdaten via syslog-Protokoll auf einem zentralen Speicher gesammelt werden.

Die Analyse im Fehlerfall eines Dienstes führen die Systemadministratoren mit typischen GNU/Linux-Werkzeugen¹⁹ durch. Durch die umfangreiche Betriebserfahrung der Administratoren ist auch mit einfachen Textanalyse-Werkzeugen eine schnelle Fehlerbehebung zu erwarten. Allerdings besteht Verbesserungspotential bei der proaktiven Auswertung von Verfügbarkeits- und Performanzdaten. Diese werden derzeit durch proprietäre und quelloffene Softwareprodukte erfasst und ausgewertet, welche aber unflexibel sind und nur schwer auf unregelmäßige Fragestellungen reagieren können.

Besonders zur spontanen Informationsgewinnung erhofft man sich Verbesserungen durch den Einsatz von Splunk. Grundsätzlich soll ein einfacherer Zugang zu den in Logdaten enthaltenen Informationen ermöglicht werden.

7.2 Bestehende Lösung an den verteilten Standorten

An den entfernten Standorten, von denen aus der DNS-Dienst erbracht wird, sind jeweils zentrale syslog-Empfänger im Einsatz. Sie sammeln pro verteiltem Standort die Logdaten der anderen dort befindlichen Linux-Systemen ein und halten sie zentralisiert vor. Dies erleichtert die Systemverwaltung von der Geschäftsstelle aus. Außerdem ist dies vorteilhaft für Statistikgeneratoren und Werkzeuge zur Überwachung der Netzwerkverfügbarkeit, die in jeder NSL im Einsatz sind.

Aufgrund des reduzierten Aufbaus einer jeden NSL müssen weniger administrative Eingriffe an den Systemen erfolgen als in der umfangreicheren RSL. Da DENIC eine 100-prozentige Verfügbarkeit des DNS-Dienstes garantiert, wird bei einem schweren Systemausfall innerhalb einer NSL der gesamte Standort aus dem globalen Internet-Routing entfernt, anstatt im laufenden Betrieb Reparaturen an einzelnen Systemen durchzuführen.

¹⁹<https://www.gnu.org/software/>

Anhand dieser Praxis wird deutlich, dass es bei NSL-Standorten auf andere Arten von Logdaten ankommt als in der RSL. Relevanter als Logdaten einzelner Systeme sind Informationen auf Ebene der Netzwerkkonnektivität und der Dienstleistung im Ganzen.

Man erhofft sich, in Bezug auf diese Anforderungen, ebenfalls von Splunk profitieren zu können. Neben dem leichteren Zugriff auf die rein technischen Informationen soll zukünftig geprüft werden, in wie weit Splunk die bisherigen Verfahren zur Statistikerfassung übernehmen oder ergänzen kann.

In bestehenden NSL-Standorten werden keine Einzelkomponenten der standardisierten Aufbauten ausgetauscht oder verändert. Stattdessen werden die Standorte unregelmäßig neu spezifiziert und als Ganzes neu aufgebaut um von zwischenzeitlichem technischem Fortschritt profitieren zu können. Sollte sich der Einsatz von Splunk in der RSL bewähren, wird die Software auch Einzug in die zukünftigen NSL-Spezifikationen halten.

7.3 Zielbild

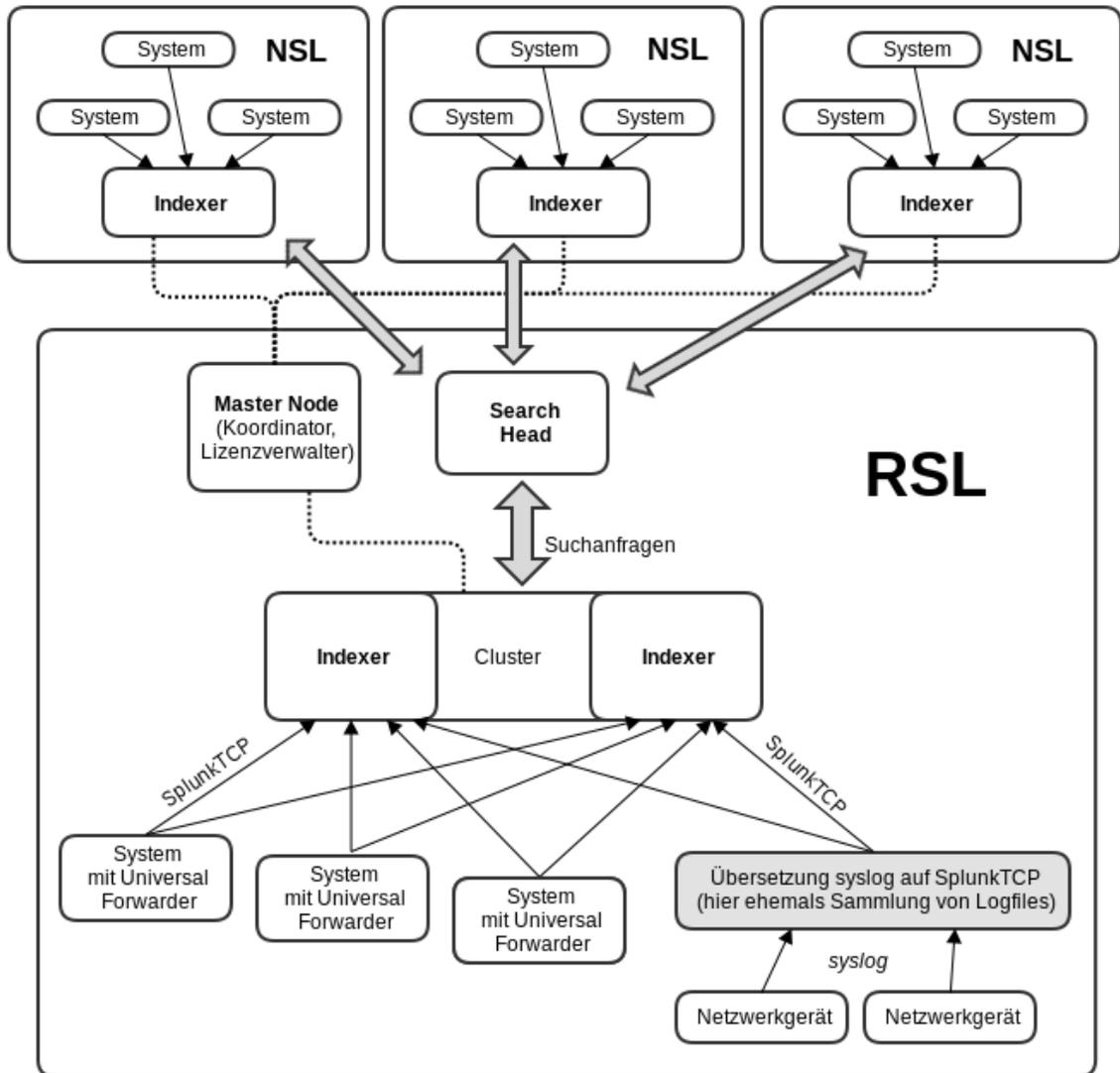


Abbildung 7.1: Gesamtschema der zu entwickelnden Infrastruktur

7.4 Implementierung am zentralen Standort

Das tägliche Logvolumen der RSL-Systeme beläuft sich auf etwa 10 GB. Diese Menge könnte mit einem Splunk-Einzelsystem bewältigt werden. Aus Gründen der Ausfallsicherheit, Skalierbarkeit und zur Abdeckung voneinander abgetrennter Netzwerksegmente wird allerdings ein Cluster aus mehreren Splunk-Systemen entwickelt.

Es wird für die folgenden Abschnitte vorausgesetzt, dass alle Systeme bereits grundlegend in das Unternehmensnetzwerk integriert sind. Dies beinhaltet fertig installierte Betriebssysteme und korrekte Netzwerk- und Firewallkonfigurationen.

7.4.1 Konfiguration des Master Node

Der erste Schritt ist die Einrichtung des Master Node für den Cluster. Hierzu wird das allgemeine Installationspaket von Splunk auf einem System installiert.

```
1 # Installationspaket mittels Paketverwaltung
2 # des Betriebssystems installieren
3 # (je nach Linux-Distribution via rpm, dpkg, yum, etc.)
4
5 # Splunk starten
6 /opt/splunk/bin/splunk start --accept-license
7
8 # Splunk in den Bootprozess einbinden
9 /opt/splunk/bin/splunk enable boot-start
```

Listing 7.1: Installation des Master Node

Danach steht die Web-Oberfläche von Splunk bereit und ist per Web-Browser auf TCP-Port 8000 des Systems zu erreichen.

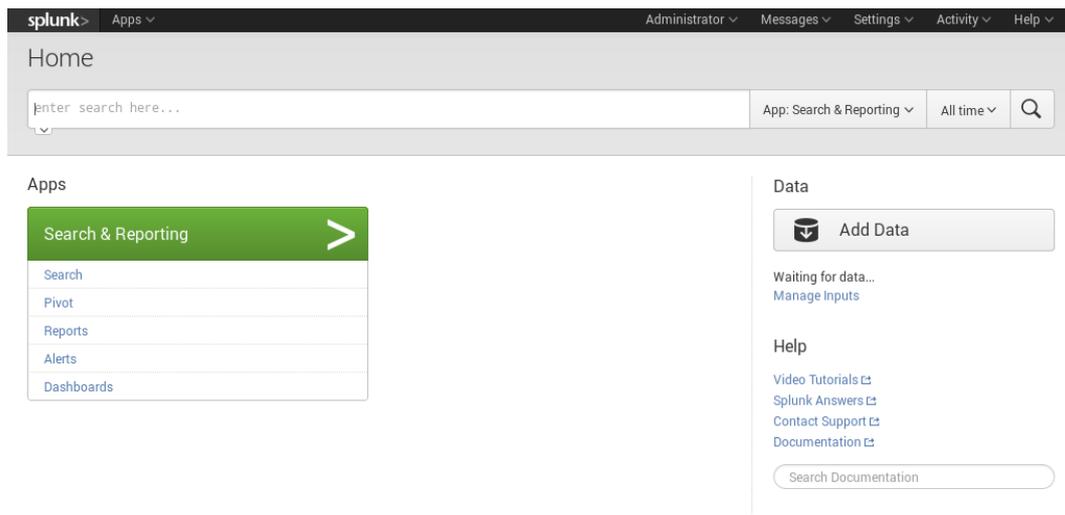


Abbildung 7.2: Startbildschirm der Splunk-Web-Oberfläche

In diesem Zustand handelt es sich um ein Splunk-Einzelsystem, das noch auf keine dedizierte Aufgabe hin konfiguriert worden ist. Um die Instanz zu einem Master Node zu machen, werden die Punkte Settings -> Clustering -> Enable Clustering der Web-Oberfläche aufgerufen. Dann folgt die Auswahl, ob das System zu einem Master Node, Peer Node (Indexer eines Clusters) oder zu einem Search Head konfiguriert werden soll. In diesem Fall wird die Option des Master Node gewählt.

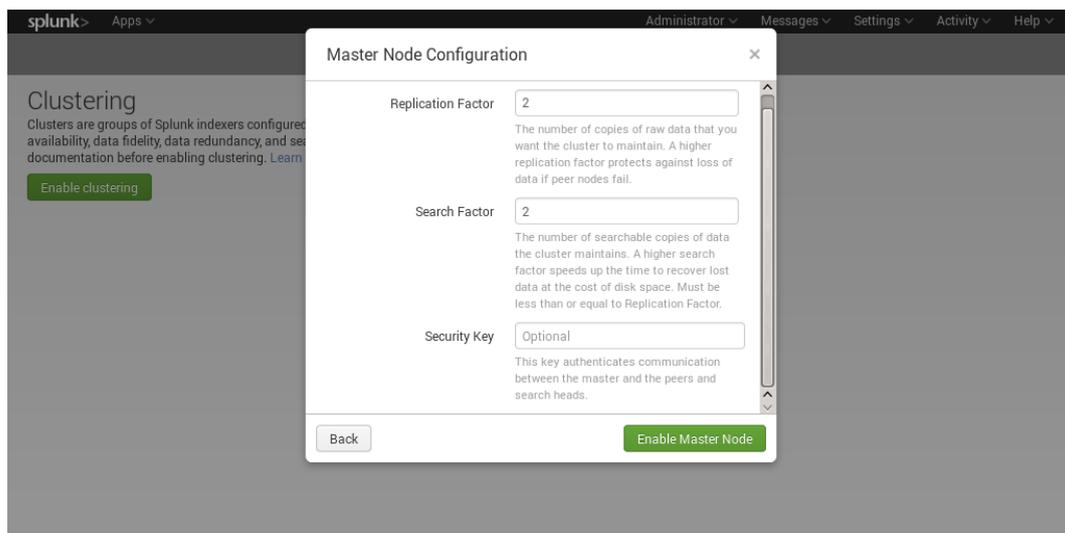


Abbildung 7.3: Konfiguration des Master Node

Die Werte `Replication Factor` und `Search Factor` bestimmen, wieviele Kopien des gesamten Datenbestand im Cluster vorgehalten sein sollen. Der erste Wert bezieht sich auf Rohdaten ohne zugehörige Indexinformationen (siehe Kapitel 5), der zweite auf komplette Datenbestandskopien mitsamt Indexinformationen. Letztere verbrauchen mehr Speicherplatz auf einem Indexer während zu indexlosen Kopien bei Zugriff erst Indexdateien generiert werden müssen.

Da im geplanten Cluster zwei Indexer existieren werden, stehen beide Werte auf 2.

Die Eingabe eines Sicherheitsschlüssels ist optional und dient zur Authentifizierung der einzelnen Komponenten innerhalb der Splunk-Infrastruktur untereinander. Ohne Sicherheitsschlüssel wäre es theoretisch denkbar, dass ein Angreifer eigene Splunk-Komponenten in die Infrastruktur integriert und Zugriffe auf Datenbestände oder Manipulationen der Infrastruktur durchführen kann.

Nach der Bestätigung des Dialogs ist der Master Node fertig konfiguriert und wartet auf sich anmeldende Indexer.

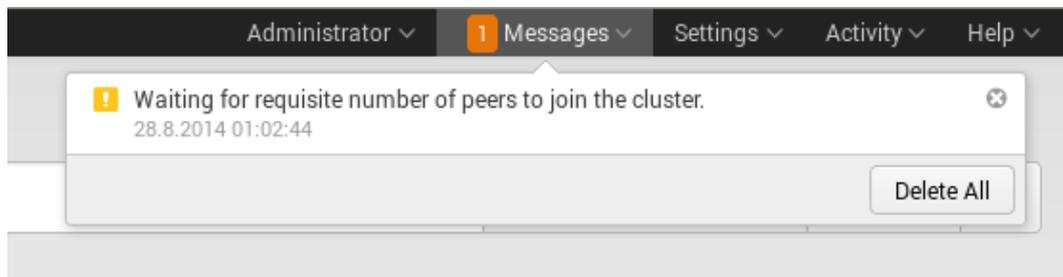


Abbildung 7.4: Mitteilung: Master Node wartet auf anmeldende Indexer

7.4.2 Konfiguration der Indexer

Die Installation ist mit der des Master Node identisch bis zum Dialog unter `Settings -> Clustering -> Enable Clustering`. Hier wird nun „Peer Node“ ausgewählt.

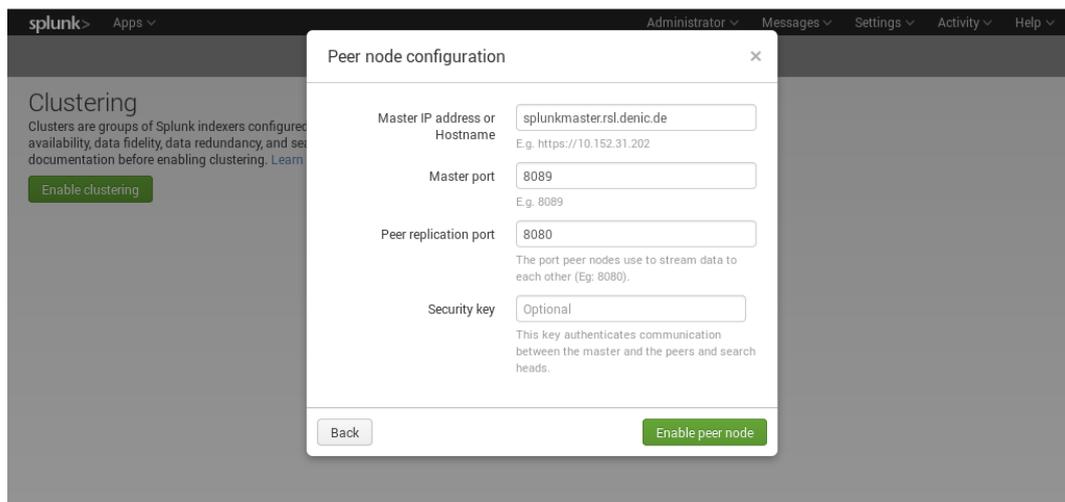


Abbildung 7.5: Konfiguration eines Indexers

Die nötigen Eingaben sind die Adresse des zuvor eingerichteten Master Nodes, der TCP-Port über welchen die Koordination des Clusters durch den Master Node erfolgt sowie ein TCP-Port über den die eigentliche Replikation der Datenbestände stattfinden soll. Falls beim Master Node ein Sicherheitsschlüssel gesetzt wurde, muss dieser im dritten Feld angegeben werden.

Nach Bestätigung des Dialogs verbindet sich der Indexer mit dem Master Node.

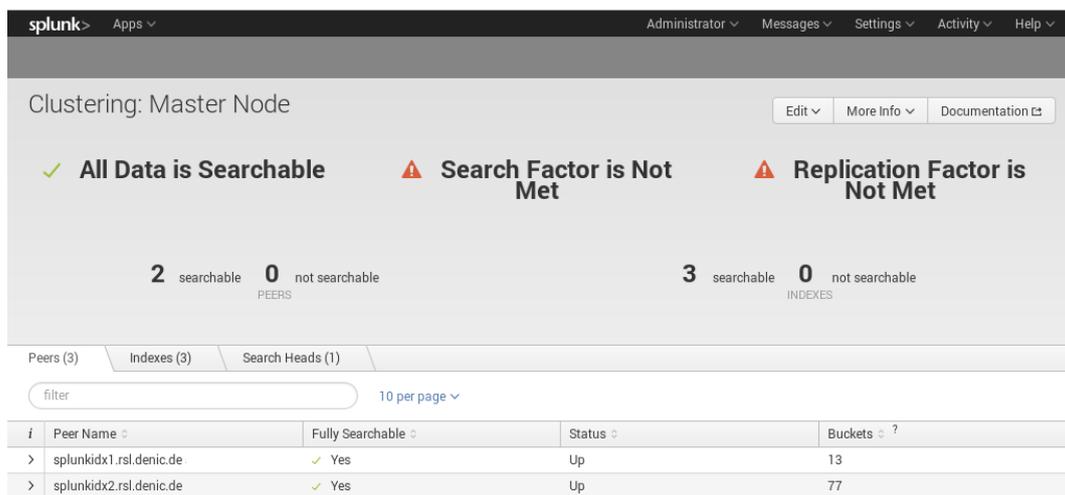


Abbildung 7.6: Am Master Node angemeldete Indexer

Die Übersicht in Abbildung 7.6 zeigt, welche Systeme aktiv den Splunk-Cluster bilden. Abzulesen ist auch deren Netzwerkstatus und ob vollständige Suchfähigkeit vorhanden ist oder nicht. Anhand der Anzahl der Buckets pro Indexer lässt sich eine grobe quantitative Einschätzung der auf dem Indexer vorhandenen Datenmengen treffen. Die wichtigsten Informationen zum Zustand des Clusters sind relativ prominent im oberen Teil der Anzeige platziert. Von links nach rechts lässt sich dort ablesen, ob der gesamte im Cluster indizierte Datenbestand durchsuchbar ist und ob die geforderten Anzahlen von indexlosen Kopien und durchsuchbaren Kopien existieren.

Im Beispiel sind die beiden letzten Kriterien noch nicht erfüllt, da die Abbildung unmittelbar nach der Anmeldung der Indexer am Master Node erstellt wurde. So ist zu sehen, dass vor dem Beginn des Cluster-Betriebs noch eine initiale Überprüfung der Cluster-Bereitschaft stattfindet.

Die Indexer verfügen zu diesem Zeitpunkt noch über keinerlei Daten-Input. Die Logdaten des Indexer-Systems selbst werden zu einem späteren Zeitpunkt integriert, zunächst wird der Empfang von über das Netzwerk eingehenden Daten eingerichtet.

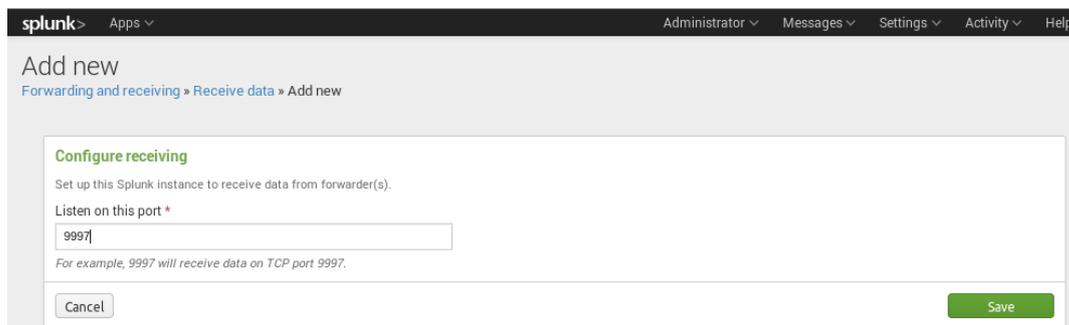


Abbildung 7.7: Indexer: Annehmen von übers Netzwerk eingehenden Daten

Nach dem Speichern wartet der Indexer auf TCP-Port 9997 (SplunkTCP) auf eingehende Datenströme von Forwardern.

Dieser Abschnitt muss für jeden Indexer, der am Cluster teilnehmen soll, wiederholt werden. Sind alle Indexer entsprechend eingerichtet, werden die Forwarder installiert.

7.4.3 Konfiguration der Forwarder

Auf allen Systemen, welche ihre Logdaten in die neue Logging-Infrastruktur einliefern sollen, wird der „Universal Forwarder“ installiert. Da in dessen Installationspaket ausschließlich die Weiterleitungsfunktionen enthalten sind, gibt es hier kein Webinterface, über das die Konfiguration erfolgt. Stattdessen wird die Einrichtung über textuelle Konfigurationsdateien vorgenommen.

```
1 # Installationspaket mittels Paketverwaltung
2 # des Betriebssystems installieren
3 #   (je nach Linux-Distribution via rpm, dpkg, yum, etc.)
4
5 # Splunk starten
6 /opt/splunk/bin/splunk start --accept-license
7
8 # Splunk in den Bootprozess einbinden
9 /opt/splunk/bin/splunk enable boot-start
10
11 # Forwarder gegen alle Indexer konfigurieren,
12 # damit load-balancing stattfinden kann
13 splunk add forward-server splunkidx1.rsl.denic.de:9997
14 splunk add forward-server splunkidx2.rsl.denic.de:9997
15 # (...)
16
17 # Definieren, welche lokalen Daten vom
18 # Forwarder weitergeleitet werden
19 #   (siehe Listing 7.3)
20 vim /opt/splunk/etc/system/local/inputs.conf
21
22 # Forwarder-Dienst an IP-Adresse statt
23 # alle Schnittstellen binden
24 #   (siehe Listing 7.4)
25 vim /opt/splunk/etc/splunk-launch.conf
26
27 # Dienst neu starten, damit alle
28 # Konfigurationsänderungen aktiv werden
29 /etc/init.d/splunk restart
```

Listing 7.2: Installation des Splunk Universal Forwarder

```
1 [default]
2 host = mail-in-7.rsl.denic.de
3
4 [monitor:///var/log]
5 index = main
```

Listing 7.3: Universal Forwarder: inputs.conf-Beispiel

Mit dem Wert von `host` (in diesem Beispiel ein Mailserver) identifiziert sich der Forwarder gegenüber den Indexern. Der folgende `monitor`-Block definiert, dass alle Dateien in `/var/log` an die Indexer gesendet werden sollen und dort in den Index `main` eingefügt werden sollen. Dies ist der Standardwert, weshalb die Zeile in diesem Fall optional ist. Wären die Daten für einen anderen Index bestimmt, würde man es hier angeben.

Das Senden von Empfangsbestätigungen, die Indexer an Forwarder zurückschicken, ist nicht standardmäßig aktiviert. Zur Aktivierung wird folgender Eintrag in der Datei `outputs.conf` auf jedem Forwarder vorgenommen.

```
1 ...
2 useACK = true
3 ...
```

Listing 7.4: Aktivieren der Indexer-Empfangsbestätigungen

Die Forwarder kommunizieren mit dem Master Node um zu erfahren, ob alle Indexer des Clusters verfügbar sind oder ob manche derzeit nicht als Ziel für ausgehenden Datenströme verwendet werden dürfen. Der Forwarder-Dienst öffnet für diese Kommunikation den TCP-Port 8089 auf allen Schnittstellen des Systems. Dies kann allerdings ungewollte Auswirkungen haben, da in der Regel mehrere Schnittstellen im System vorhanden sind und nicht garantiert ist, dass der Port 8089 nicht schon auf einer anderen belegt ist. In diesem Fall würde der Forwarder nicht starten können.

Diese potentielle Fehlerquelle wird mit einem Eintrag in der Datei `splunk-launch.conf` behoben.

```
1 ...
2 SPLUNK_BINDIP=10.4.4.7
3 ...
```

Listing 7.5: Änderung der Schnittstelle für Master-Kommunikation

Nach einem Neustart läuft der Forwarder nur noch auf der IP-Adresse `10.4.4.7` anstatt auf allen verfügbaren.

Weitere Aktionen sind an dieser Stelle nicht notwendig. Insbesondere müssen die bisher auf dem System bestehenden Logging-Mechanismen nicht abgeschaltet oder verändert werden, da der Universal Forwarder die regulären Logdateien des Systems unter `/var/log` ja gerade als Datenquelle verwendet.

Die Schritte dieses Abschnitts müssen für alle Systeme wiederholt werden, die mittels eines Forwarders in die Splunk-Infrastruktur hinein loggen sollen.

Es mag hier der Eindruck entstehen, die Konfiguration eines Forwarders (ausgerechnet der am öftesten zu installierenden Komponente) wäre die umfangreichste von allen Komponenten. Dies ist allerdings ein Trugschluss, bedenkt man die Tatsache, dass ausschließlich dateibasierte Konfigurationen anstelle einer Web-Oberfläche genutzt wurden. Dadurch ist es leicht, Forwarder durch Verwaltungswerkzeuge wie „Chef“²⁰, „Puppet“²¹ oder „CFEngine“²² vollautomatisch auf beliebig vielen Systemen installiert zu bekommen.

7.4.4 Sonderfall: Netzwerkhardware

In der bestehenden Lösung bilden Netzwerkgeräte eine Ausnahme im Logverhalten. Sie halten ihre Logdateien nicht selbst vor, sondern leiten sie via syslog-Protokoll an einen Sammelserver. Dieser hat zwar einen

²⁰<http://www.getchef.com/>

²¹<https://www.puppetlabs.com/>

²²<http://www.cfengine.org/>

Universal Forwarder erhalten und ist in die Splunk-Infrastruktur integriert, allerdings beachtet dessen Forwarder bisher nur das Verzeichnis `/var/log`. Die gesammelten Logdateien der Netzwerkgeräte liegen an einem anderen Ort des Dateisystems und werden vom Forwarder bisher nicht erfasst.

Es gibt nun drei Möglichkeiten, auch die Logs der Netzwerkhardware in Splunk einzuliefern.

Forwarder liest zusätzliche Logdateien

Die einfachste Möglichkeit ist, die Forwarder-Konfiguration um einen `monitor`-Eintrag bezüglich des zusätzlichen Verzeichnisses zu erweitern (siehe Listing 7.3). Der Vorteil dieser Lösung ist, dass effektiv nur eine neue Zeile zur Konfiguration nötig ist und keine weiteren Änderungen am Logging der Netzwerkgeräte durchgeführt werden müssen.

Übergabe von syslog an Forwarder

Anstelle des `monitor`-Eintrages bezüglich des zusätzlichen Verzeichnisses könnten die via syslog am Sammelsystem eintreffenden Daten auch direkt an den Forwarder weitergereicht werden, ohne in lokalen Dateien niedergeschrieben zu werden. Diese Variante ist mit der vorherigen weitgehend identisch, spart allerdings den Speicherplatz der Logdaten auf dem Sammelsystem ein.

In der Konfigurationsdatei `inputs.conf` des Forwarders auf dem Sammelsystems wäre folgende Erweiterung nötig.

```
1 [udp://514]
2 connection_host=ip
3 sourcetype=syslog
```

Listing 7.6: Universal Forwarder: `inputs.conf` für Sammelsystem

Der bisher laufende syslog-Dienst müsste nun dauerhaft abgeschaltet werden, da er sonst den UDP²³-Port 514 besetzt und der Splunk-Forwarder nicht gestartet werden könnte.

²³User Datagram Protocol

Netzwerkgeräte direkt an Splunk-Indexer anbinden

Diese Möglichkeit ist nur der logischen Vollständigkeit halber genannt. Da Netzwerkgeräte verschiedener Hersteller untereinander inkompatible Zugriffs- und Steuerungsmöglichkeiten anbieten, kann keine automatisierte Konfigurationsänderung aller Netzwerkgeräte erfolgen. Eine massenhafte, manuelle Konfiguration aller unterschiedlichen Geräteklassen und -modelle widerspricht hier dem Anspruch, mit modernen und effizienten Techniken zu arbeiten.

Die zweite Variante erscheint als empfehlenswerteste, da sie geringen Konfigurationsaufwand erfordert, künftig keinen weiteren Speicherplatz auf dem Sammelsystem mehr benötigt und nicht von einem Ausfall sekundärer Festplatten im Sammelsystem beeinträchtigt wäre.

7.4.5 Konfiguration des Search Heads

An dieser Stelle werden bereits die Logdaten aller Systeme mit Forwardern inklusive der Logdaten der Netzwerkgeräte im Cluster indiziert. Um Suchanfragen durchführen zu können wird nun der Search Head installiert, der den Benutzern hierzu eine Web-Oberfläche bereitstellt. Ähnlich wie bei Master Node und Indexern ist auch ein Search Head eine allgemeine Splunk-Installation, die anschließend zum Search Head erklärt wird.

Die Installation des Search Head ist wieder bis zum Dialog Settings -> Clustering -> Enable Clustering identisch mit der Installation von Master Node und Indexern.

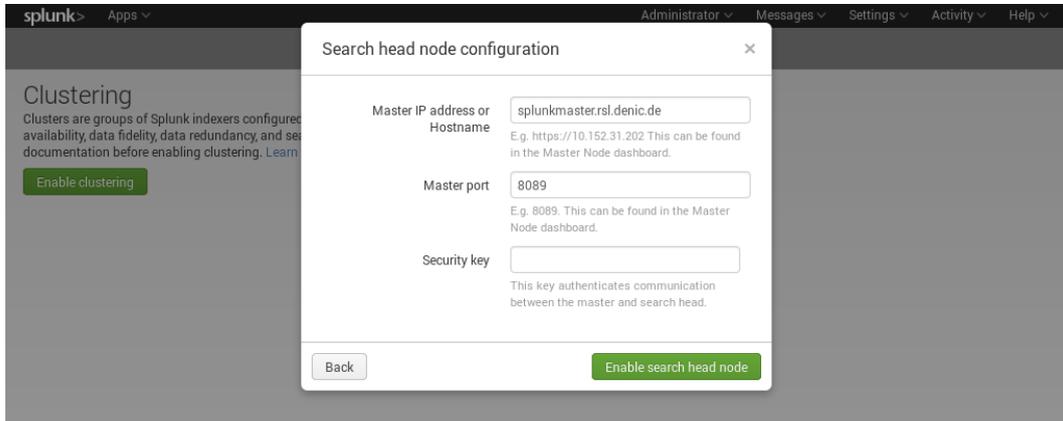


Abbildung 7.8: Konfiguration des Search Head

Nach der Bestätigung des Dialogs ist der Search Head fertig installiert.

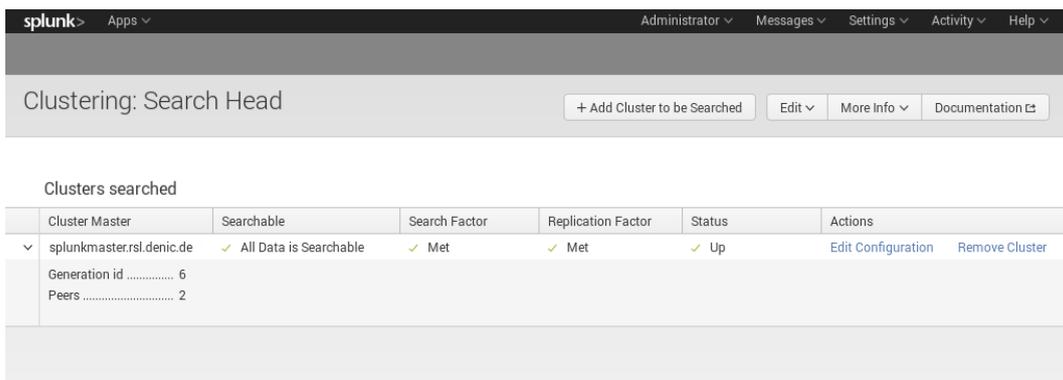


Abbildung 7.9: Search Head: Verbundener Cluster

In Abbildung 7.9 ist ersichtlich, dass der Search Head zum Splunk-Cluster verbunden ist und dass dieser korrekt arbeitet. In Umgebungen mit mehreren Clustern könnten nun auch diese eingetragen werden. Der Search Head ist also nicht auf einen Cluster beschränkt.

7.5 Konfiguration cluster-lokaler Datenquellen

Die Splunk-Infrastruktur am zentralen Unternehmensstandort ist nun beinahe vollständig implementiert. Die einzigen Systeme, die noch keine selbst produzierten Logdaten in den Cluster einliefern sind Master Node, Search Head und die Indexer selbst. In den jeweiligen Web-Oberflächen wird ein lokaler Dateninput angelegt.

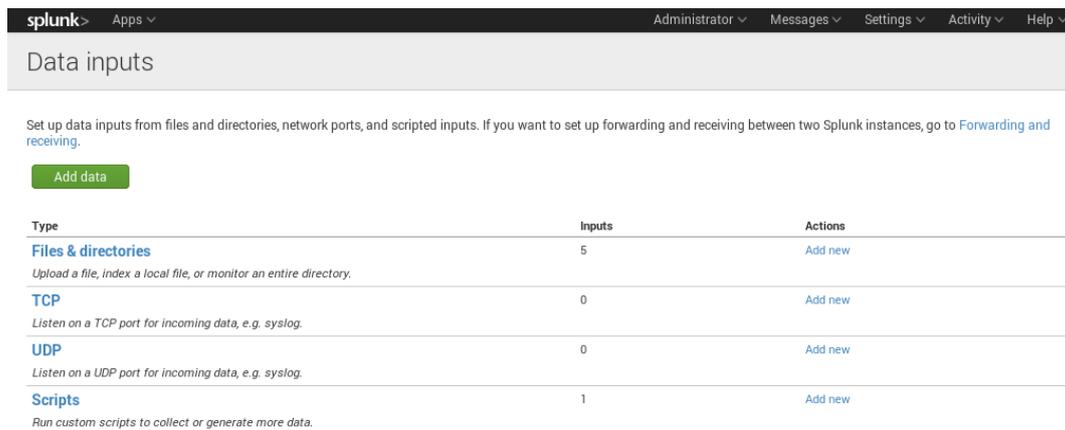


Abbildung 7.10: Verfügbare Datenquellen

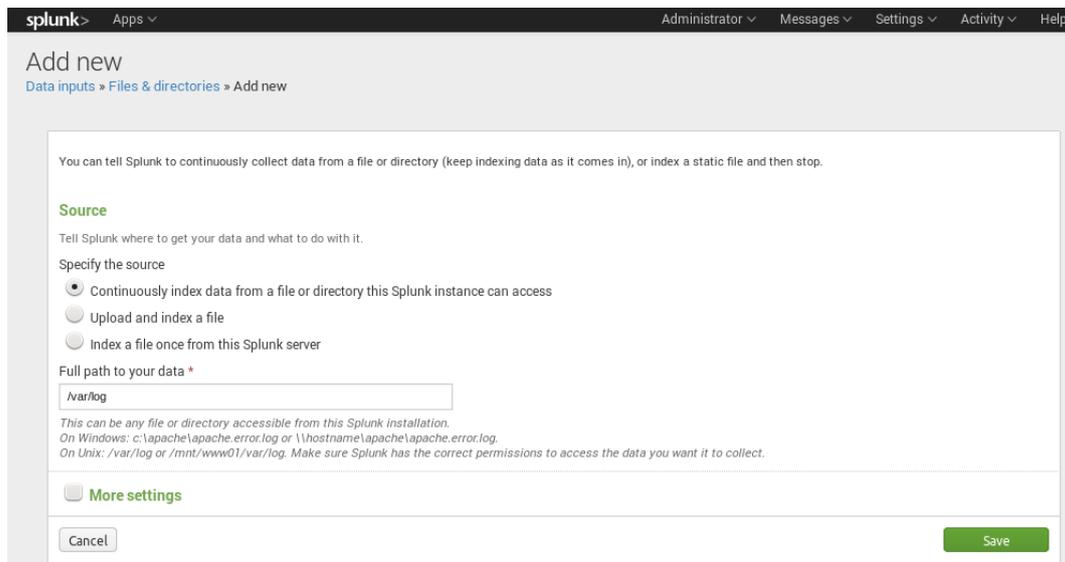


Abbildung 7.11: Einlesen Indexer-lokaler Logdaten

Das Eintragen des Log-Verzeichnisses über die Web-Oberfläche äußert sich auf Ebene der Konfigurationsdateien genau wie bei den Forwardern: Die lokale `inputs.conf` erhält einen `monitor`-Eintrag bezüglich `/var/log/` (siehe Listing 7.3).

Im Falle der Indexer sind keine weiteren Schritte nötig. Ihre lokalen Logdaten werden eingelesen und indiziert, da sie sich ja schon direkt am Ziel befinden. Da Master Node und Search Head ihre Logdaten allerdings nicht selbst verarbeiten sollen, werden diese noch angewiesen, die eingelesenen Daten an die eigentlichen Indexer des Clusters weiterzuleiten. Dazu werden entsprechende Forwards über die jeweilige Web-Oberfläche eingetragen.

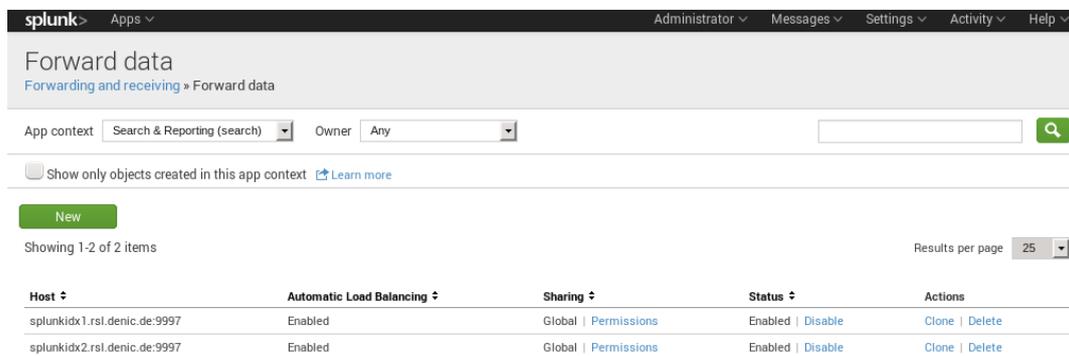


Abbildung 7.12: Forwards für Master Node und Search Head

7.6 Konfiguration der Lizenzen

Um den Cluster unter konsistenten Lizenzen (näheres in Kapitel 10) zu betreiben, wird der Master Node als lizenzverwaltendes System des Clusters konfiguriert. Ihm wird über die Web-Oberfläche (Licensing -> Change License Group -> Enterprise License) eine „Enterprise“-Lizenz zugewiesen, die den vollen Funktionsumfang des Clusters erlaubt.

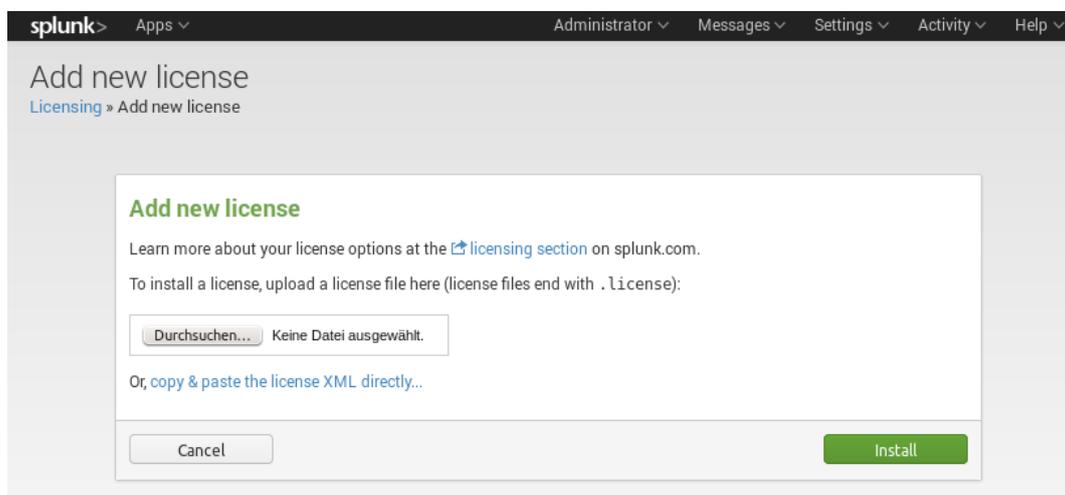


Abbildung 7.13: Zuweisen einer Enterprise-Lizenz

Dem Search Head und den Indexern wird, ebenfalls über die jeweiligen Web-oberflächen, unter Licensing -> Change to slave der Master Node als lizenzverwaltendes System bekannt gemacht. Bis dies geschieht, ist jedes dieser Systeme noch sein eigener Lizenzverwalter und verfügt per Installation über eine nach 60 Tagen auslaufende Testlizenz.

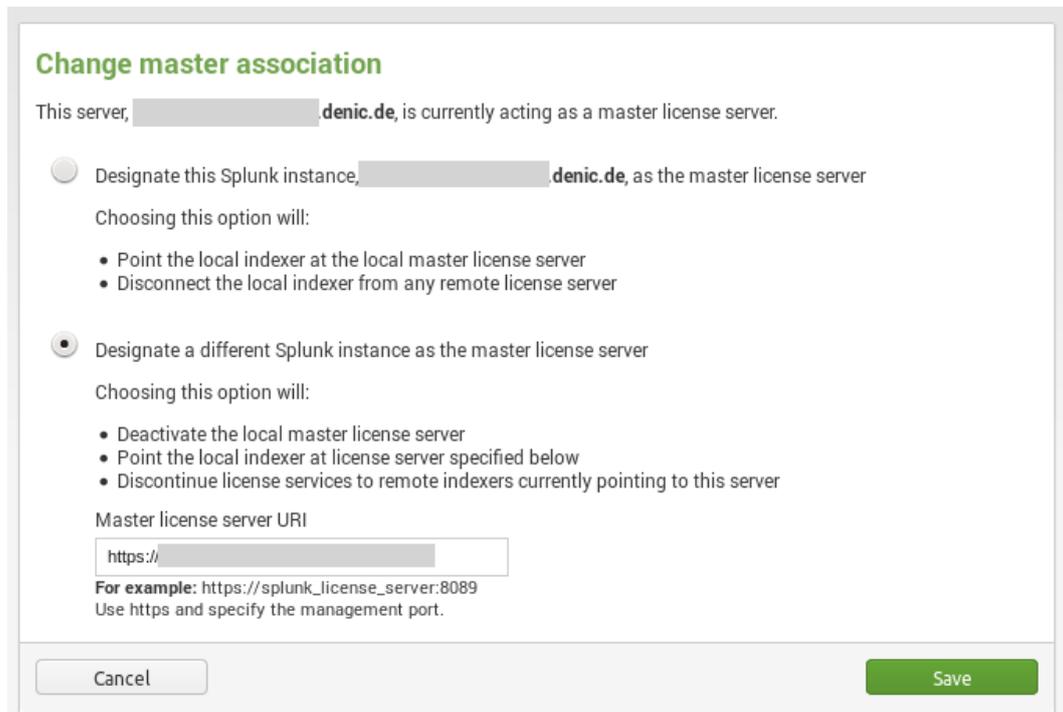


Abbildung 7.14: Festlegung des Master Nodes als Lizenzverwalter

Für die Forwarder sind keine weiteren Schritte notwendig. Sie haben bei der Installation automatisch eine spezielle Forwarder-Lizenz erhalten. Diese ist kostenlos und nicht zeitlich begrenzt.

Nachdem Search Head und Indexer gegen den Master Node als Lizenzverwalter konfiguriert wurden, ist der Cluster am zentralen Standort vollständig implementiert und benutzbar.

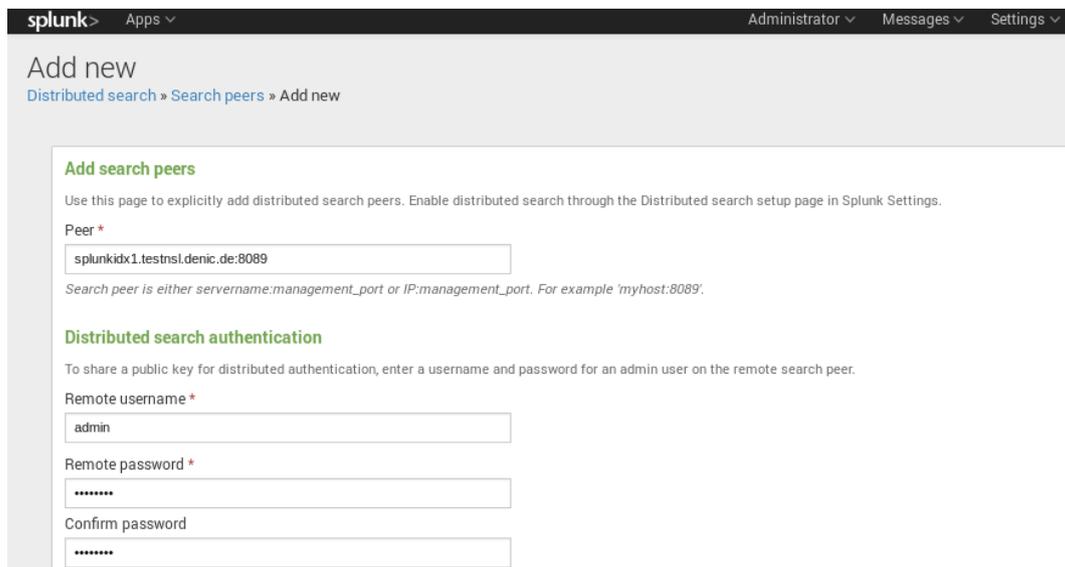
7.7 Integration der entfernten Standorte

Die Integration von beliebig vielen entfernten Standorten besteht nun nur noch aus der Wiederholung einiger zuvor beschriebener Schritte.

An einem entfernten Standort muss

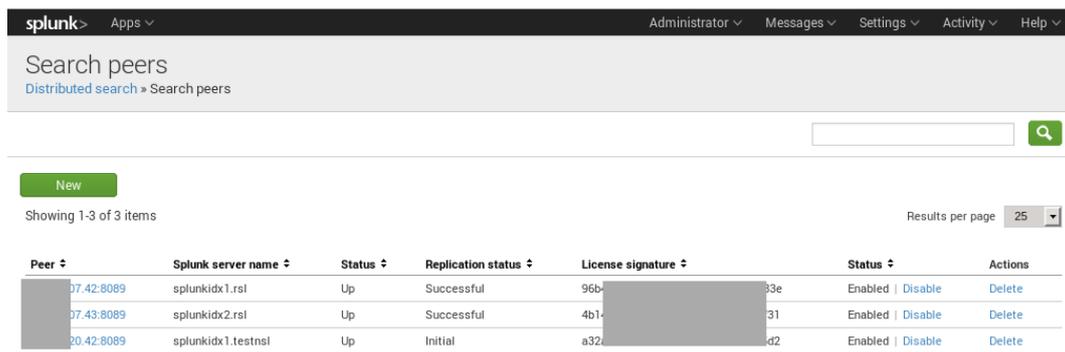
- jeweils ein Indexer installiert werden (siehe Abschnitt 7.4.2),
- ein Forwarder auf allen anderen System installiert werden (siehe Abschnitt 7.4.3),
- der Indexer konfiguriert werden, seine selbst generierten Logdaten zu erfassen (siehe Abschnitt 7.5) und
- dem Indexer der Master Node am zentralen Standort als Lizenzverwalter zugewiesen werden (siehe Abschnitt 7.6).

Da der Indexer des entfernten Standorts kein Teil des Clusters ist, wird sein Datenbestand bisher auch nicht vom Search Head berücksichtigt. Einem Search Head können aber nicht nur Cluster sondern auch einzelne Indexer zugewiesen werden. Die Zuweisung erfolgt über die Web-Oberfläche des Search Heads (Distributed Search -> Search Peers).



The screenshot shows the Splunk web interface. At the top, there is a navigation bar with 'splunk>' and 'Apps' on the left, and 'Administrator', 'Messages', and 'Settings' on the right. Below the navigation bar, the main content area is titled 'Add new' with a breadcrumb trail: 'Distributed search > Search peers > Add new'. The main content area is divided into two sections. The first section is titled 'Add search peers' and contains the following text: 'Use this page to explicitly add distributed search peers. Enable distributed search through the Distributed search setup page in Splunk Settings.' Below this text is a form field labeled 'Peer *' with the value 'splunkidx1.testnsl.denic.de:8089'. Below the form field is a note: 'Search peer is either servername:management_port or IP:management_port. For example 'myhost:8089''. The second section is titled 'Distributed search authentication' and contains the following text: 'To share a public key for distributed authentication, enter a username and password for an admin user on the remote search peer.' Below this text are three form fields: 'Remote username *' with the value 'admin', 'Remote password *' with a masked password '*****', and 'Confirm password' with a masked password '*****'.

Abbildung 7.15: Search Head: Entfernten Standort integrieren



The screenshot shows the Splunk web interface for 'Search peers'. At the top, there is a navigation bar with 'splunk>' and 'Apps' on the left, and 'Administrator', 'Messages', 'Settings', 'Activity', and 'Help' on the right. Below the navigation bar, the page title is 'Search peers' with a breadcrumb 'Distributed search > Search peers'. A search input field is visible on the right. Below the search bar, there is a 'New' button and the text 'Showing 1-3 of 3 items'. On the right side, there is a 'Results per page' dropdown set to '25'. The main content is a table with the following columns: Peer, Splunk server name, Status, Replication status, License signature, Status, and Actions. The table contains three rows of data.

Peer	Splunk server name	Status	Replication status	License signature	Status	Actions
07.42.8089	splunkidx1.rsl	Up	Successful	96b	33e	Enabled Disable Delete
07.43.8089	splunkidx2.rsl	Up	Successful	4b1	31	Enabled Disable Delete
20.42.8089	splunkidx1.testnsl	Up	Initial	a32	d2	Enabled Disable Delete

Abbildung 7.16: Übersicht bekannter Search Peers

Nach Bestätigung des Dialogs kennt der Search Head den Cluster sowie die einzelnen Indexer der verteilten Standorte. Vom Search Head gestartete Suchvorgänge werden an alle Search Peers verteilt und die Suchergebnisse aller Standorte konsolidiert.

Sind alle Standorte gemäß den vorherigen Abschnitten eingerichtet, die Splunk-Komponenten untereinander verbunden und lizenziert worden, ist der Splunk-Cluster fertig implementiert.

Kapitel 8

Anwendungsbeispiele

Es folgen einige Anwendungsbeispiele, welche die Search Processing Language, Suchen und Reports demonstrieren.

8.1 Suchen

Bei der Search Processing Language handelt es sich um eine Anweisungssprache zur Formulierung von Suchkriterien und Weiterverarbeitung der Suchergebnisse. Die Sprache erinnert entfernt an SQL und bietet, ähnlich einer Unix-Shell, Prozedurverkettung mittels dem `|`-Zeichen („Pipe“-Symbol) an. Sie erlaubt die logische Verknüpfung von Kriterien sowie die Benutzung von regulären Ausdrücken.

Die einfachste Suchanfrage lautet `"*`". Sie würde den gesamten Datenbestand des `main-Index` anzeigen. Für sinnvolle Suchabfragen muss die Menge der zurückgelieferten Ergebnisse nach Kriterien eingeschränkt werden. Das folgende Listing gibt eine Übersicht zu beispielhaften Suchanfragen.

```
1 # Alle Datensätze im Datenbestand auflisten, die
2 # den Begriff "error" enthalten.
3
4 "error"
5
6 # Alle Treffer im Datenbestand auflisten, die den
7 # Begriff "error" oder "warning" enthalten.
8
9 "error" OR "warning"
10
11 # Alle Treffer im Datenbestand auflisten, die den
12 # Begriff "error" und "warning" enthalten. Das
13 # logische UND ist dabei implizit.
14
15 "error" "warning"
16
17 # Alle Treffer im Datenbestand auflisten, die den
18 # Begriff "error" oder "warning", aber nicht "info"
19 # oder "debug" enthalten.
20
21 ("error" OR "warning") NOT ("info" OR "debug")
22
23 # Alle Treffer anzeigen, die aus Dateien namens
24 # /var/log/mail.log stammen
25
26 source=/var/log/mail.log
27
28 # Kombination von Quelldatei und Suchtext
29
30 source=/var/log/mail.log "relay access denied"
31
32 # Weitergeben der Suchergebnisse an die Funktion
33 # timechart, welche die Ergebnisse als Diagramm
34 # darstellt. "count by sender" bewirkt, dass im
35 # Diagramm die Häufigkeiten der einzelnen Werte
36 # des Feldes "sender" pro Zeiteinheit (Tage im
37 # Normalfall) dargestellt werden.
38
39 source=/var/log/mail.log "denied" | timechart count by sender
40
41 # Eine komplexere Suchanfrage mit regulären Ausdrücken
42 # und statistischer Aufbereitung der Ergebnisse.
43
44 process=postfix host=mail-*
45 | rex ".*?(?<exception>(?:\w+\.)+\w*?Exception).*"
46 | stats count(_raw) by exception
```

Listing 8.1: Beispiele der Search Processing Language (SPL)

Die folgende Abbildung zeigt die Benutzeroberfläche mit Suchergebnissen.

The screenshot displays the Splunk Search & Reporting interface. At the top, the search bar contains the query `source=*jboss.log` and shows 36 events from 8/7/14 12:00:00.000 AM to 9/6/14 1:51:21.000 AM. Below the search bar is a visualization area with a bar chart showing event counts over time. The main results table is displayed below, showing a list of events with their timestamps and details.

i	Time	Event
>	9/4/14 2:22:06.765 AM	02:22:06,765 INFO [stdout] (http--172.16.0.5-8080-1) [EL Info]: 2014-09-04 02:22:06.765--ServerSession(1729486400)--EclipseLink, version: Eclipse Persistence Services - 2.3.1.v20111018-r10243 date_hour = 2 date_mday = 4 date_minute = 22 date_month = september date_second = 6 date_wday = thursday date_year = 2014 index = main linecount = 1 source = /home/jboss/jboss.log sourcetype = jboss
>	9/4/14 1:32:42.515 AM	01:32:42,515 INFO [org.jboss.as.connector] (MSC service thread 1-3) JBAS010408: Starting JCA Subsystem (JBoss IronJacamar 1.0.9.Final) 01:32:43,694 INFO [org.jboss.as.connector.subsystems.datasources] (ServerService Thread Pool -- 27) JBAS010404: Deploying non-JDBC-compliant driver class com.mysql.jdbc.Driver (version 5.1) 01:32:43,889 INFO [org.jboss.as.naming] (MSC service thread 1-1) JBAS011802: Starting Naming Service date_hour = 1 date_mday = 4 date_minute = 32 date_month = september date_second = 42 date_wday = thursday date_year = 2014 index = main linecount = 3 source = /home/jboss/jboss.log sourcetype = jboss
>	9/4/14 1:32:41.216 AM	01:32:41,531 INFO [org.jboss.as.security] (MSC service thread 1-1) JBAS013100: Current PicketBox version=4.0.7.Final date_hour = 1 date_mday = 4 date_minute = 32 date_month = september date_second = 41 date_wday = thursday date_year = 2014 index = main linecount = 1 source = /home/jboss/jboss.log sourcetype = jboss
>	9/4/14 1:32:41.216 AM	01:32:41,216 INFO [org.jboss.as.configadmin] JBAS016200: Activating ConfigAdmin Subsystem 01:32:41,233 INFO [org.jboss.remoting] JBoss Remoting version 3.2.3.GA 01:32:41,269 INFO [org.jboss.as.security] JBAS013101: Activating Security Subsystem 01:32:41,277 INFO [org.jboss.as.naming] JBAS011800: Activating Naming Subsystem 01:32:41,283 INFO [org.jboss.as.clustering.infinispan] JBAS010280: Activating Infinispan subsystem. Show all 8 lines date_hour = 1 date_mday = 4 date_minute = 32 date_month = september date_second = 41 date_wday = thursday date_year = 2014 index = main linecount = 8 source = /home/jboss/jboss.log sourcetype = jboss
>	9/4/14 1:32:40.132 AM	01:32:40,132 INFO [org.jboss.as.server] JBAS015888: Creating http management service using socket-binding (management-http) 01:32:40,347 INFO [org.xnio] XNIO Version 3.0.3.GA 01:32:40,622 INFO [org.xnio.nio] XNIO NIO Implementation Version 3.0.3.GA date_hour = 1 date_mday = 4 date_minute = 32 date_month = september date_second = 40

Abbildung 8.1: Web-Oberfläche mit Suchergebnissen

Das Bereich `Visualization` stellt spontane Visualisierungen der Suchergebnisse bereit. Im Folgenden wird eine solche an einem simplen Beispiel gezeigt.

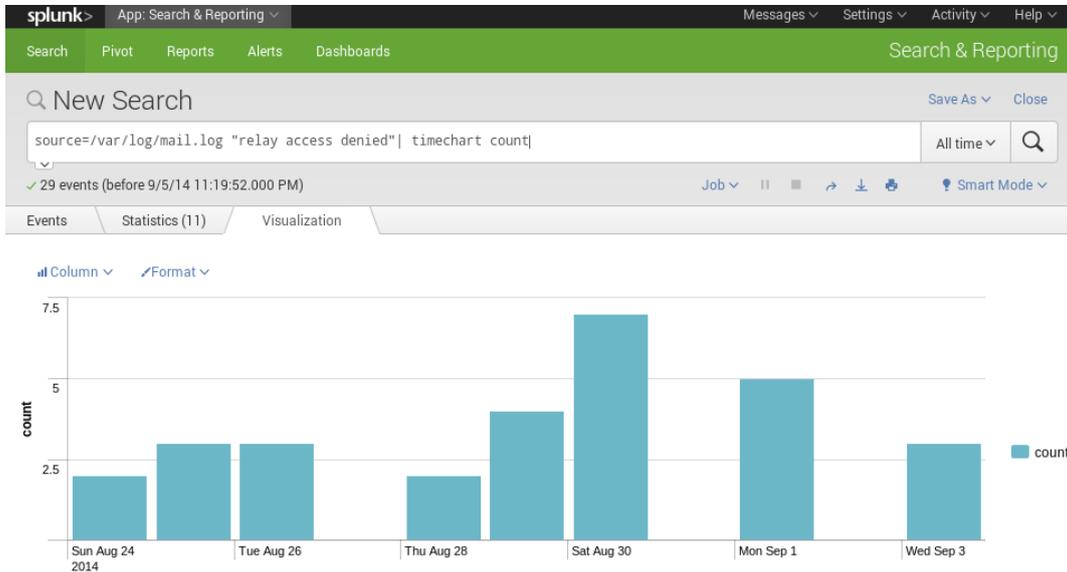


Abbildung 8.2: Visualisierung von Suchergebnissen

Dargestellt ist hier die Häufigkeit der Meldung über eine abgelehnte E-Mail im Verlauf einer Woche. Der Menüpunkt `Format` erlaubt die spontane Umgestaltung des Diagrammtyps zu einem Linien-, Flächen- oder Kreisdiagramm.

8.2 Reports und Dashboards

Über den Menüpunkt `Save As -> Report` kann die aktuelle Suche mit samt ihrer Ergebnisse und den Visualisierungen gespeichert werden. Eine gespeicherte, betiteltete Suche heißt „Report“. Auf ihre Ergebnisse kann durch das Aufrufen des Reports später erneut zugegriffen werden. Ein Report kann auch automatisch gemäß einem Zeitplan ausgeführt werden. Die Ergebnisse werden nach jedem Lauf per E-Mail versandt. Außerdem kann die Visualisierung eines Reports auf einem „Dashboard“ platziert werden. Dashboards sind Flächen, auf denen mehrere verschiedene Visualisierungen angeordnet sind und kontinuierlich dargestellt werden.

Es folgen zwei Beispiele von Dashboards. Diese könnten bei DENIC auf großen Monitoren in den Räumen der Administratoren dargestellt werden. Sie könnten auf einen Blick Aufschluss über den momentanen Status bestimmter Dienste liefern.

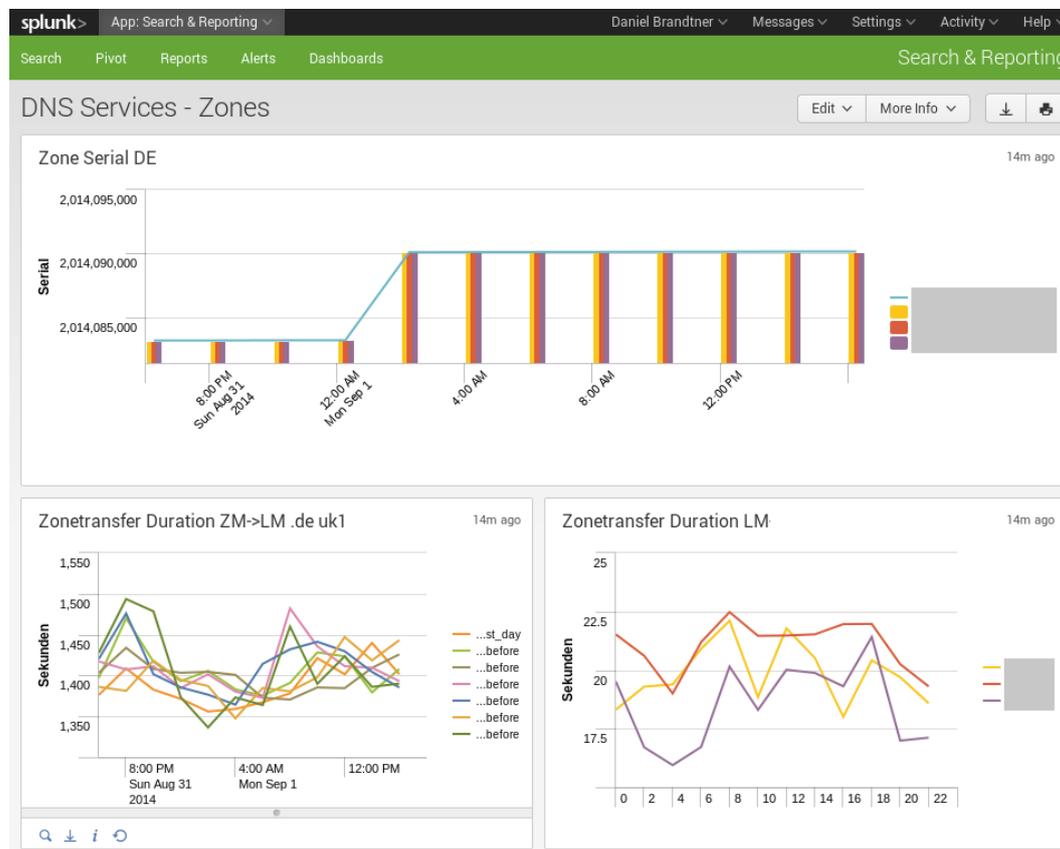


Abbildung 8.3: Dashboard mit Reports

Auf dem Dashboard in Abbildung 8.3 befinden sich Informationen bezüglich der DNS-Dienste. Sollten bei der Datenverteilung zwischen den entfernten Standorten Fehler auftreten, wäre dies sofort erkennbar. Im obersten der drei Reports wären die farbigen Säulen dann nicht mehr gleich hoch, da die Versionsnummer der letzten erfolgreich erhaltenen Datensätze vertikal dargestellt ist.

Die unteren beiden Reports eignen sich zur schnellen Erkennung von Transferverzögerungen. Wiederkehrende Muster mit hohen Werten zu bestimmten Uhrzeiten beispielsweise können an diesen Liniendiagrammen bequem erkannt werden.

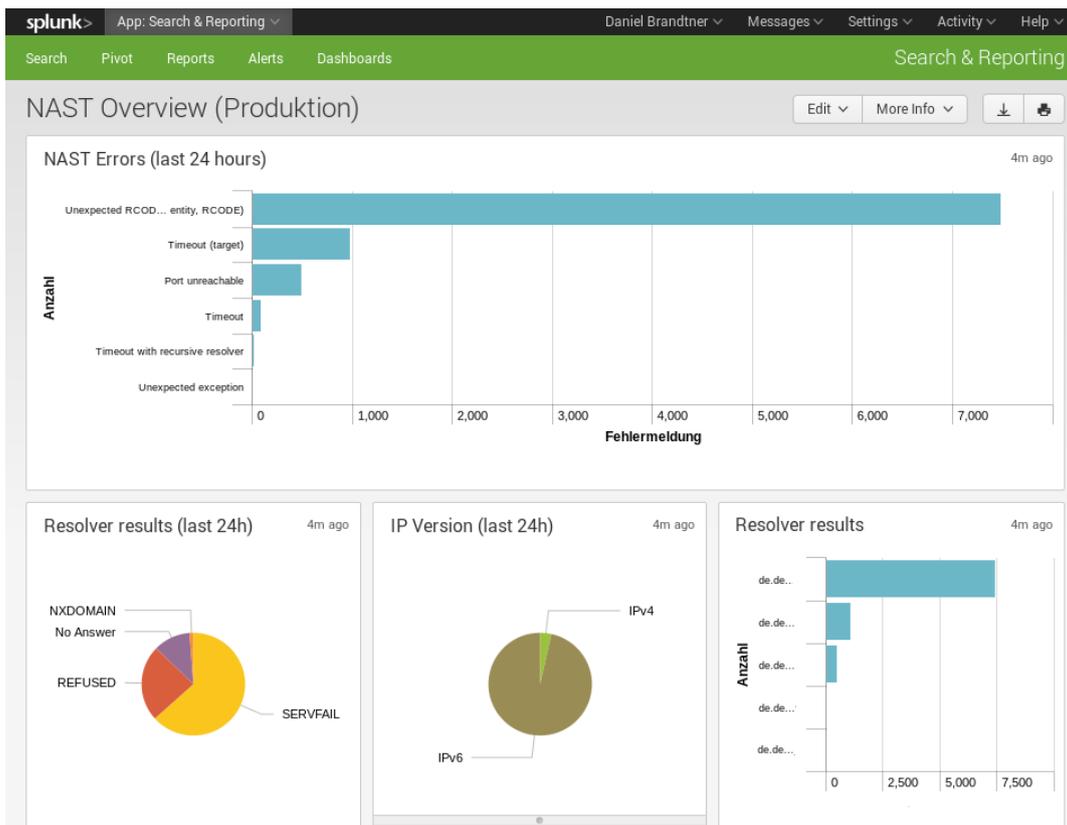


Abbildung 8.4: Dashboard mit Reports

Auf dem Dashboard in Abbildung 8.4 sind die Häufigkeiten von fehlerhaften Auftragstypen an einen Dienst abzulesen. Die Kreisdiagramme geben Aufschluss über den Anteil von IPv4- beziehungsweise IPv6-Klienten und die Verteilung von Antwortarten über alle Aufträge.

Kapitel 9

Sicherheit

In diesem Kapitel werden einige Techniken zur Härtung von Splunk in Bezug auf Vertraulichkeit, Verfügbarkeit und Integrität der Daten beschrieben.

9.1 Verfügbarkeit

Durch Cluster-Betrieb lässt sich eine hohe Verfügbarkeit erzielen. Durch die vom Replication Factor eines Clusters bestimmte Anzahl der vollständigen Kopien innerhalb eines Clusters können bis zu (*Replication Factor - 1*) Indexer ausfallen, ohne dass die Datenbestände unzugreifbar werden. Sollten mehr Indexer gleichzeitig ausfallen als der Search Factor des Clusters beträgt, müssen erst vorhandene indexlose Kopien des Datenbestands durchsuchbar gemacht werden. [8]

Durch Search Head Pooling ergibt sich eine ähnliche Redundanz auf Benutzerebene. Hier können alle bis auf einen Search Head ausfallen, bevor es zu Einschränkungen des Betriebs kommt.

9.2 Authentizität

Um die Authentizität von Daten zu gewährleisten müssen diese vor Manipulationen durch Unbefugte geschützt werden. Hierzu können Zugriffsberechtigungen auf Basis von Benutzerkonten oder Benutzerklassen definiert werden.

Nutzer- und Rollenkonzept

Splunk verfügt über ein klassisches Benutzer- und Rollenkonzept. Dies bedeutet dass pro Benutzer individuell bestimmt werden kann, welche Aktionen er ausführen darf oder nicht. Beispiele für solche einschränkbaren Aktionen sind Suchen auf bestimmten Indizes, das Löschen von Datenbeständen, das Verwalten von Indizes oder das dauerhafte Speichern von Suchvorgängen.

Berechtigungen können auch an Rollen statt an Benutzer gekoppelt werden. Auf diese Weise muss nicht für jeden einzelnen Benutzer die Menge von erlaubten Aktionen definiert werden, da diese über die Rollenzugehörigkeiten des Benutzers an ihn vererbt werden. Dies ermöglicht es beispielsweise in einem Unternehmen die Rollen „Admin“ (alle Rechte), „Support“ (nur Leserechte auf bestimmten Indizes) und „Data Research“ (Leserechte überall) zu definieren. Durch das Hinzufügen der Mitarbeiterkonten zu den Rollen würden die jeweiligen Mitarbeiter die ihrer Rolle entsprechenden Berechtigungen erhalten.

Benutzerkonten können entweder manuell in der Splunk-Benutzerdatenbank eingerichtet werden oder aus einem LDAP-Pool²⁴ ausgelesen werden. Falls das Unternehmen bereits eine Mitarbeiterverwaltung auf LDAP-Basis pflegt, kann sich Splunk dieser Informationen bedienen. Manuell müsste dann nur noch die Definition von Rollen und die Zuordnung von LDAP-Benutzern zu den Rollen erfolgen. [17]

²⁴Lightweight Directory Access Protocol

9.3 Vertraulichkeit

Die Vertraulichkeit der Daten sicherzustellen bedeutet, sie zum Beispiel durch Transportverschlüsselung vor dem Zugriff durch Unbefugte zu schützen.

Verschlüsselungsfunktionen sind standardmäßig nicht bei allen Komponenten aktiviert, da Splunk in der Regel in internen, durch ohnehin starke Schutzmaßnahmen abgesicherten Netzwerken eingesetzt wird. Das zusätzliche Verschlüsseln aller Querverbindungen per Splunk-eigenem SSL beispielsweise würde in einem abgeschotteten Szenario keinen Sicherheitsgewinn, wohl aber zusätzlichen Aufwand für die Systeme und die vorhandene Infrastruktur bedeuten.

Welche zusätzlichen Sicherheitsfunktionen für das jeweilige Einsatzszenario aktiviert werden sollten, muss individuell betrachtet werden.

Secure Socket Layer (SSL)

Für die Browserkommunikation mit den Web-Oberflächen der einzelnen Komponenten kann SSL im entsprechenden Untermenü (System Settings -> General Settings -> Enable SSL) eingeschaltet werden.

Bei einer Splunk-Installation werden SSL-Zertifikate generiert, die aber nur von einer inoffiziellen Splunk-CA²⁵ unterzeichnet sind und somit Sicherheitswarnungen in den verbreiteten Webbrowsern auslösen. Um dies zu vermeiden, müssen andere, offizielle SSL-Zertifikate auf allen Splunk-Komponenten hinterlegt werden.

Die interne Kommunikation der Komponenten wie beispielsweise von Forwardern zu Indexern oder innerhalb eines Clusters kann ebenfalls SSL-verschlüsselt werden. Dies kann aber nicht über die Web-Oberfläche geschehen, sondern muss über die Konfigurationsdateien jeder Komponente durchgeführt werden. [17]

²⁵Certificate Authority

9.4 Integrität

Schutz der Datenintegrität bedeutet, die Bestände vor Zerstörung oder Verfälschung durch technische Fehlfunktionen zu schützen.

Empfangsbestätigungen

Die Integrität der indizierten Datenbestände kann durch Empfangsbestätigungen beim Forwarding unterstützt werden. Der praktische Nutzen ist, dass Forwarder ihre zu versendenden Daten nicht als abgearbeitet markieren, solange kein Indexer den korrekten Empfang signalisiert hat. Dies ist ein Vorteil gegenüber dem klassischen syslog-Protokoll beispielsweise, das weder selbst, noch in tiefer liegenden Netzwerkschichten solche Bestätigungen vorsieht. Bei Netzwerkproblemen oder Schwierigkeiten auf dem Empfangssystem, sind einmal an der Quelle versandte Daten verloren.

Integrität der Indizes

Splunk verfügt über eigene Mechanismen um die Integrität seiner Datenbestände auf Bucket-Basis zu überwachen. Sollten Buckets als fehlerhaft erkannt werden, werden automatische Reperaturversuche unternommen und bei ausbleibendem Erfolg die Administratoren informiert. Über Reperaturwerkzeuge kann dann versucht werden, die beschädigten Buckets zu reparieren oder sie bei schweren Fehlern manuell zu entfernen. [8]

Kapitel 10

Lizenzierung

Das Lizenzierungsmodell von Splunk unterscheidet sich von klassischen Modellen für kommerzielle Software. Üblicherweise wird eine Software für eine bestimmte Anzahl von Systemen oder Benutzern lizenziert. Bei Splunk ist die Anzahl der Systeme und Benutzer unerheblich, relevant ist ausschließlich das Volumen der indizierten Rohdaten pro Tag. [16]

Im Cluster-Betrieb mit mehreren Indexern übernimmt in der Regel der Search Head zusätzlich die Aufgabe der Lizenzverwaltung. Er überwacht die Summe des an allen Indexern eingelieferten Datenvolumens und gibt Warnungen aus, sollte das lizenzierte Volumen überschritten werden. Durch die Einrichtung von Lizenzpools kann die Zuteilung von erlaubten Volumen zu einzelnen Indexern verwaltet werden. Auf diese Weise können beispielsweise auch mehrere Standorte mit unterschiedlich großen Logging-Aufkommen unterhalb der selben Lizenz betrieben werden.

Die Preise für die verschiedenen Volumenlizenzen sind nicht öffentlich einsehbar. Die Kundenbetreuung von Splunk Inc. erstellt auf Anfrage individuelle Angebote, die sich nach den konkreten Anforderungen des Kunden richten. Die Firma gibt jedoch auf ihren Webseiten²⁶ an, dass 1 GB tägliches Volumen für etwa \$1.800 pro Jahr lizenziert werde (Stand August 2014). Größere Lizenzen beinhalten Indizierungsvolumen von 10 oder 100 GB täglich. Diese können für ein Jahr gelten oder unbefristet gültig sein.

²⁶<http://www.splunk.com/view/how-to-get-splunk/SP-CAAADFV>

10.1 Lizenzverletzung

Im Falle einer Überschreitung des täglich erlaubten Indizierungsvolumens erfolgt eine Warnung an die Administratoren. Es besteht nun die Möglichkeit, die lizenzverletzende Situation durch das Installieren einer anderen, ausreichend großen Lizenz auflösen. Die Situation kann auch durch veränderte Volumenzuweisungen im Lizenzpool aufgelöst werden, sofern an anderer Stelle noch genug freies Volumen vorhanden ist.

Erfolgt keine Auflösung bis Mitternacht des selben Tages, wird die vorläufige Warnung bezüglich der Verletzung als permanent aufgezeichnet. Bei mehr als fünf solcher permanenter Einträge wird die Suchfunktionalität der Splunk-Infrastruktur abgeschaltet bis eine ausreichende Lizenz eingespielt wird. Von der Abschaltung ist niemals die Indizierungsfunktionalität selbst betroffen, sodass trotz Lizenzverletzung keine Unterbrechung der infrastrukturellen Verfügbarkeit eintritt. [16]

splunk > Apps ▾

Licensing

This server is acting as a master license server [Change to slave](#)

Enterprise license group

[Change license group](#)

This server is configured to use licenses from the **Enterprise license group**

[Add license](#) [Usage report](#)

Alerts

Licensing alerts notify you of excessive indexing warnings and licensing misconfigurations. [Learn more](#)

Current

- No licensing alerts

Permanent

- No licensing violations

Splunk Enterprise stack

[Learn more](#)

Licenses	Volume	Expiration	Status
Splunk Enterprise	10,240 MB	Jan 19, 2038 4:14:07 AM	valid
Effective daily volume	10,240 MB		

Pools	Indexers	Volume used today
auto_generated_pool_enterprise		1 MB / 9,216 MB Edit Delete
	.denic.de	0 MB (0%)
	.denic.de	1 MB (>0%)
	.denic.de	0 MB (0%)
	.denic.de	0 MB (0%)
NSL		0 MB / 1,024 MB Edit Delete
	.denic.de	0 MB (0%)

[+ Add pool](#)

Local server information

Indexer name [redacted].denic.de

Volume used today 0 MB

Warning count 0

Debug information [All license details](#)
[All indexer details](#)

Abbildung 10.1: Lizenzpool-Übersicht unter Enterprise-Lizenz

Kapitel 11

Fazit

In dieser Bachelor-Thesis wurden die grundlegenden Funktionen von Splunk behandelt, die Rahmenbedingungen und Komponenten erläutert und ein voll funktionsfähiger Cluster konzipiert. Dieser übertrifft nach Einschätzung des Autors bereits in seiner rudimentären Form die Fähigkeiten von klassischen Logging-Methoden in verteilten Netzwerkstrukturen.

Dennoch konnte der volle Funktionsumfang von Splunk im Rahmen und Umfang dieser Bachelor-Thesis nur angeschnitten werden. Viele weitere Funktionen, gerade im Bereich der mehrdimensionalen Datenzusammenführung, fortgeschrittenen Visualisierung und kontinuierlichen Überwachung von Ereignissen haben letztendlich keinen Einzug in diese Arbeit gefunden. Diese Funktionen zum Thema eigenständiger Bachelor-Arbeiten zu machen, ist ohne Weiteres denkbar.

Sofern die hier konzipierte Infrastruktur tatsächlich bei DENIC eingesetzt und weiterentwickelt wird, werden die Mitarbeiter im Laufe der Zeit zweifellos Nutzen aus den bisher noch unberücksichtigten Funktionen ziehen können und diese implementieren.

11.1 Vergleich der Plattformen

Ein direkter Vergleich zu den alten Logging-Verfahren wird dadurch erschwert, dass die neue Infrastruktur die vorherige nicht ersetzt, sondern erweitert. Die Logdateien der vorher bestehenden Methoden bleiben schließlich ohne jede Veränderung erhalten und dienen der Splunk-Infrastruktur als Datenquellen. (Die einzige Ausnahme bildet hier das ausgemusterte Netzwerkgeräte-Logging der RSL, welches keine festplattenbasierten Logdateien mehr schreibt.)

Es ist jedoch offensichtlich, dass ein gewisser Nachteil in den Lizenzkosten von Splunk liegt, wohingegen die vorher bestehenden Werkzeuge und Verfahren kostenfrei beziehungsweise unter Open Source-Lizenzen einzusetzen waren.

Abgesehen von wirtschaftlichen Faktoren wie Lizenzen und zusätzlich benötigter Hardware stellt sich die neue Infrastruktur ausschließlich vorteilhaft dar. Die mitunter deutlichsten Verbesserungen sind sicherlich die Möglichkeiten zur Analyse und Aufbereitung der Datenbestände, die nur noch einen Bruchteil des vorherigen Aufwands erfordern sowie die Vereinigung von unixoiden und nicht-unixoiden Betriebssystemen auf der selben Logging-Plattform.

11.2 Ausblick

Der konzipierte Cluster ist horizontal frei skalierbar. Sollten sich in Zukunft *erhebliche* Steigerungen des Logging-Volumens abzeichnen, können dem RSL-Cluster einfach weitere Indexer hinzugefügt werden. Bei erheblich steigendem Logging-Volumen in den verteilten Standorten können deren Einzel-Indexer auch gegen Cluster ausgetauscht werden. Zu diesem Zeitpunkt sollte jedoch eine neue Evaluierung stattfinden, ob inzwischen neue Techniken bezüglich Multiclustering in Splunk integriert wurden.

Durch die umfangreiche Konfigurierbarkeit kann auch zukünftigen Anforderungen Sorge getragen werden, auf die man keinen direkten Einfluss hat. Beispielsweise können sich Logformate externer Software ändern oder

neue datenschutzrechtliche Verhältnisse in Bezug auf Datenspeicherung und Vorhaltefristen entstehen. Die Filter- und Anonymisierungstechniken von Splunk könnten in diesem Fall genutzt werden, um den geltenden Bedingungen weiterhin gerecht zu werden.

Diese Überlegungen sind natürlich rein hypothetischer Natur. Es ist nicht abzusehen, ob sich beispielsweise das Lizenzmodell von Splunk dahingehend ändern wird, dass ein weiterer Einsatz aus wirtschaftlichen oder lizenzrechtlichen Gründen unattraktiv wird.

Vom rein technischen Standpunkt aus und in Bezug auf die reell nur sehr langsam ansteigenden Logvolumina ist in den kommenden Jahren aber keinerlei Erweiterungs- oder Skalierungsbedarf der hier vorgeschlagenen, neuen Infrastruktur zu erwarten. Die Investition in Lizenzen und die produktive Implementierung von Splunk Enterprise wird DENIC empfohlen.

Abbildungsverzeichnis

2.1	syslog gegenüber direktem Schreiben	8
2.2	Anwendungsbeispiel syslog-facilities	10
2.3	Rotation von Textdateien	12
4.1	Schematische Darstellung des Splunk-Indexers	24
4.2	Schematische Darstellung des Universal Forwarder	26
4.3	Schematische Darstellung des Heavy Forwarder	27
4.4	Schematische Darstellung eines Clusters mit Search Head . . .	28
6.1	Einzelner Splunk-Server	37
6.2	Mehrere separate Indexer	38
6.3	Cluster mit Lastverteilung	39
6.4	Splunk-Infrastruktur über mehrere Standorte	41
6.5	Schematische Darstellung eines Clusters mit Search Head Pool	42
7.1	Gesamtschema der zu entwickelnden Infrastruktur	46
7.2	Startbildschirm der Splunk-Web-Oberfläche	48
7.3	Konfiguration des Master Node	48
7.4	Mitteilung: Master Node wartet auf anmeldende Indexer . . .	49
7.5	Konfiguration eines Indexers	50
7.6	Am Master Node angemeldete Indexer	50
7.7	Indexer: Annehmen von übers Netzwerk eingehenden Daten	51

7.8	Konfiguration des Search Head	57
7.9	Search Head: Verbundener Cluster	57
7.10	Verfügbare Datenquellen	58
7.11	Einlesen Indexer-lokaler Logdaten	58
7.12	Forwards für Master Node und Search Head	59
7.13	Zuweisen einer Enterprise-Lizenz	60
7.14	Festlegung des Master Nodes als Lizenzverwalter	61
7.15	Search Head: Entfernten Standort integrieren	62
7.16	Übersicht bekannter Search Peers	63
8.1	Web-Oberfläche mit Suchergebnissen	66
8.2	Visualisierung von Suchergebnissen	67
8.3	Dashboard mit Reports	68
8.4	Dashboard mit Reports	69
10.1	Lizenzpool-Übersicht unter Enterprise-Lizenz	76

Tabellenverzeichnis

2.1	Einige facilities des syslog-Standards	9
2.2	Severity levels des syslog-Standards	10
2.3	Testergebnisse des Kompressionstests	14
4.1	Systemanforderungen	25
5.1	Kriterien für Bucketrotation	35

Literaturverzeichnis

- [1] "The GNU C Library - 18.1 Overview of Syslog." https://www.gnu.org/software/libc/manual/html_node/Overview-of-Syslog.html, 2001. [Online; Zugriff am 12.08.2014].
- [2] "Filesystem Hierarchy Standard." <http://www.pathname.com/fhs/>, 2004. [Online; Zugriff am 12.08.2014].
- [3] "SANS Institute - The Ins and Outs of System Logging Using Syslog." <https://www.sans.org/reading-room/whitepapers/logging/ins-outs-system-logging-syslog-1168>, 2003. [Online; Zugriff am 15.08.2014].
- [4] "The GNU C Library - 18.2.2 syslog, vsyslog." https://www.gnu.org/software/libc/manual/html_node/syslog_003b-vsyslog.html#syslog_003b-vsyslog, 2001. [Online; Zugriff am 12.08.2014].
- [5] D. Carasso, *Exploring Splunk*. New York, NY, USA: CITO Research, 2012, ISBN 978-0-9825506-7-0.
- [6] "Splunk Enterprise Documentation: Knowledge Manager Manual." <http://docs.splunk.com/index.php?title=Documentation:Splunk:Knowledge:WhatIsSplunkknowledge:6.0beta&action=pdfbook>. [Online; Zugriff am 28.08.2014].
- [7] "Splunk Enterprise Documentation: Getting Data In." <http://docs.splunk.com/index.php?title=Documentation:Splunk:Data:WhatSplunkcanmonitor:6.0beta&action=pdfbook>. [Online; Zugriff am 20.08.2014].

- [8] "Splunk Enterprise Documentation: Managing Indexers and Clusters." <http://docs.splunk.com/index.php?title=Documentation:Splunk:Indexer>Aboutindexesandindexers:6.0&action=pdfbook>. [Online; Zugriff am 12.08.2014].
- [9] "Splunk Enterprise Documentation: Search Manual." <http://docs.splunk.com/index.php?title=Documentation:Splunk:Search:Whatsinthismanual:6.0beta&action=pdfbook>. [Online; Zugriff am 28.08.2014].
- [10] "Splunk Enterprise Documentation: Dashboards and Visualizations." <http://docs.splunk.com/index.php?title=Documentation:Splunk:Viz>Aboutthismanual:Cupcake&action=pdfbook>. [Online; Zugriff am 04.09.2014].
- [11] V. Bumgarner, *Implementing Splunk*. Birmingham, UK: PACKT Publishing, 2013, ISBN 978-1-84969-328-8.
- [12] "Splunk Enterprise Documentation: Search Reference." <http://docs.splunk.com/index.php?title=Documentation:Splunk:SearchReference:WhatsInThisManual:6.0beta&action=pdfbook>. [Online; Zugriff am 03.09.2014].
- [13] "Splunk Enterprise Documentation: Installation Manual." <http://docs.splunk.com/index.php?title=Documentation:Splunk:Installation:Whatsinthismanual:6.0&action=pdfbook>. [Online; Zugriff am 12.08.2014].
- [14] "Splunk Enterprise Documentation: Forwarding Data." <http://docs.splunk.com/index.php?title=Documentation:Splunk:Forwarding>Aboutforwardingandreceivingdata:6.0&action=pdfbook>. [Online; Zugriff am 17.08.2014].
- [15] "Splunk Enterprise Documentation: Distributed Search." <http://docs.splunk.com/index.php?title=Documentation:Splunk:DistSearch:Whatisdistributedsearch:6.0&action=pdfbook>. [Online; Zugriff am 12.08.2014].
- [16] "Splunk Enterprise Documentation: Admin Manual." <http://docs.splunk.com/index.php?title=Documentation:Splunk:>

Admin:Howtousethismanual:6.0beta&action=pdfbook. [Online; Zugriff am 07.09.2014].

- [17] "Splunk Enterprise Documentation: Securing Splunk." <http://docs.splunk.com/index.php?title=Documentation:Splunk:Security:WhatyoucansecurewithSplunk:5.0beta&action=pdfbook>. [Online; Zugriff am 29.08.2014].