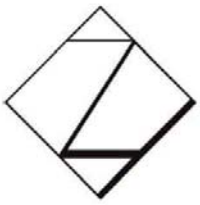

Westsächsische Hochschule Zwickau
University of Applied Sciences
Fakultät Elektrotechnik

DIPLOMARBEIT

Evaluierung von Open-Source Monitoring Systemen und Integration eines geeigneten Systems in eine Videoverarbeitungskette im digitalen Broadcasting-Bereich zur Analyse des Datenflusses.

zur Erlangung des

akademischen Grades

Diplom-Ingenieur für Informationstechnik (FH)

eingereicht von

Daniel Schmidt

geb. am 19. August 1988

Hochschullehrer:	Prof. Dr.-Ing. habil. Christian Troll
Auftraggeber oder Firma:	Rohde & Schwarz GmbH & Co. KG
Betreuer:	Dipl.-Ing. Axel Voss

Inhaltsverzeichnis

Verzeichnis der Bilder

Verzeichnis der Tabellen

Kurzzeichenverzeichnis

N	Note
N_{Max}	Maximalnote
N_{Min}	Mindestnote
P	Punkte
P_{Max}	Maximalpunkte
P_{Min}	Mindestpunkte
t_A [h]	Ausfallzeit
t_B [h]	Gesamtbetriebszeit
t_I [s]	Intervall
V [%]	Verfügbarkeit
vps $[1/s]$	Messwert pro Sekunde
x	Punkt auf Abszissenachse

Verzeichnis verwendeter Abkürzungen

API	Application Programming Interface
ASI	Asynchronous Serial Interface
CIM	Common Information Model
CSV	Comma-Separated Values
DBMS	Datenbankmanagementsystem
DDP	Datagram Delivery Protocol
DMTF	Distributed Management Task Force
DTS	Decoding Timestamp
DVB	Digital Video Broadcasting
DVB-C	Digital Video Broadcasting - Cable
DVB-S	Digital Video Broadcasting - Satellite
DVB-T	Digital Video Broadcasting - Terrestrial
ES	Elementarströme
FAQ	Frequently Asked Questions
FCAPS	Fault, Configuration, Accounting, Performance, Security
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
IP	Internet Protokoll
IPC	Interprozesskommunikation
IPFIX	Internet Protocol Flow Information Export
ISP	Internet Service Provider

JSON-RPC	JavaScript Object Notation Remote Procedure Call
MIB	Management Information Base
NOC	Network Operations Center
NSCA	Nagios Service Check Acceptor
OID	Object Identifier
PAT	Program Association Table
PCM	Puls-Code-Modulation
PCR	Program Clock Reference
PDU	Protocol Data Unit
PES	Packetized Elementary Stream
PID	Package ID
PMT	Program Map Table
PSI	Program Specific Information
PTS	Presentation Timestamp
RCA	Root Cause Analysis
RFC	Request for Comments
RRD	Round-Robin Datenbank
RTP	Real-Time Transport Protocol
SDI	serielles digitales Interface
SGMP	Simple Gateway Management Protocol
SHM	Shared Memory
SLA	Service-Level-Agreement
SMI	Structure of Management Information
SNMP	Simple Network Management Protocol
SSH	Secure Shell
STC	System Time Clock
TCP	Transmission Control Protocol
TS	Transportstrom
UDP	User Datagram Protocol
WBEM	Web-Based Enterprise Management

Glossar

Asynchronous Serial Interface (ASI)

Das ASI ist eine Schnittstelle für die Übertragung von Fernsehinhalten im digitalen Kabelnetz eines Transportstromes. Dabei wird jedes Byte des Stromes auf 10 Bit erweitert und unabhängig von der Datenrate des Transport Stream mit einem festen Bit-Takt von 270 MHz übertragen. Die feste Datenrate wird durch Hinzunahme von Fülldaten ohne Informationsinhalt erzielt.

Decoding Timestamp (DTS)

Der DTS ist ein Wert im PES-Header und repräsentiert den Dekodier-Zeitpunkt des betreffenden PES-Pakets. Dieser Zeitstempel ist nur dann vorhanden, wenn er vom Presentation Timestamp (PTS) abweicht. Das ist bei Videoströmen der Fall, wenn Differenzbilder übertragen werden und somit die Reihenfolge der Dekodierung nicht mit der Reihenfolge der Ausgabe übereinstimmt.

Elementary Stream (ES)

Ein Elementarstrom ist der Oberbegriff für codiertes Video, Audio und andere codierten Bitströme. Er beschreibt den Datenstrom von MPEG-2 mit den komprimierten digitalen Audio- und Videosignalen.

MPEG-2

MPEG-2 ist ein Normenwerk (ISO/IEC 13818), welches die Codierung und Komprimierung von Bild (Teil 2) und Ton (Teil 3) zum jeweiligen Elementarstrom beschreibt sowie die Zusammenführung der Elementarströme zu einem Transportstrom durch die Multiplexbildung (Teil 1).

Packetized Elementary Stream (PES)

Nach der Digitalisierung und Kompression wird der „endlose“ Elementarstrom für die Übertragung in Pakete unterteilt. Dieser Datenstrom wird als Packetized Elementary Stream bezeichnet.

Payload

Mit Payload wird das Nutzdatenvolumen in Datenpaketen bezeichnet. Bezogen auf einen Elementarstrom sind dies nur die Nutzdaten des betreffenden Elementarstromes ohne den PES-Header.

Presentation Timestamp (PTS)

Der Presentation Timestamp ist ein Wert im PES-Header und repräsentiert den Ausgabezeitpunkt des Inhalts eines PES-Pakets.

Serial Digital Interface (SDI)

Unter Serial Digital Interface (SDI) gibt es mehrere verabschiedete Standards für eine serielle Schnittstelle zur Übertragung von unkomprimiertem Digitalvideo über Koaxialkabel. Der SDI-Standard spezifiziert mehrere Übertragungsraten für Composite Video und Komponentenvideo in normaler und hoher Auflösung im Normal- und Breitbildformat.

Shared Memory (SHM)

Shared Memory ist ein vom Betriebssystem verwalteter Speicherbereich, der von mehreren Prozessen gelesen und beschrieben werden kann. Hierzu ist eine Synchronisation der einzelnen Prozesse untereinander erforderlich, um einen gleichzeitigen Zugriff der Prozesse zu vermeiden.

Simple Network Management Protocol (SNMP)

Das Simple Network Management Protocol (SNMP) erlaubt ein zentrales Netzwerkmanagement für verschiedene Netzwerkkomponenten und beschreibt den Aufbau der Datenpakete sowie den Kommunikationsablauf.

Transport Stream (TS)

Der Transport Stream ist ein von MPEG-2 definierter Multiplexdatenstrom, der mehrere Programme enthalten kann, die wiederum jeweils aus mehreren Elementarströmen bestehen können. Die Multiplexbildung geschieht durch die Bildung von 188 Byte großen TS-Paketen für jeden Elementarstrom und die Aneinanderreihung dieser von verschiedenen Elementarströmen stammenden TS-Pakete.

YCbCr

Das YCbCr-Farbmodell besteht aus dem Luminanzsignal (Y), das die Helligkeit repräsentiert und zwei Chrominanzsignalen für die Buntheit. Das digitalisierte Cb-Signal entspricht der Farbabweichung Blau-Gelb und das digitalisierte Cr-Signal steht für die Farbabweichung Rot-Türkis.

Einleitung

Das Fernsehen gehört als Massenmedium zusammen mit dem Internet zu den Leitmedien der heutigen Gesellschaft. Dabei existieren die Grundlagen der analogen Fernsehtechnik bereits seit über 100 Jahren. Im Bereich der Fernsehstudios werden seit Anfang der 90er Jahre digitale Datensignale für die Verteilung und Verarbeitung von Fernsehsignalen eingesetzt. Gleichzeitig wurden im Rahmen des „Digital Video Broadcasting“ (kurz: DVB) Projekts mehrere digitale Verfahren zur Übertragung der Fernsehsignale über Satellit (DVB-S), über Koaxialkabel (DVB-C) sowie auf terrestrischem Wege (DVB-T) entwickelt. Diese Übertragungsverfahren bieten pro Kanal Bruttodatenraten zwischen 12 und 20 MBit/s für DVB-T sowie 38 Mbit/s für DVB-S bzw. DVB-C. Da die Datenrate der unkomprimierten Fernsehsignale im Studiobereich bei circa 270 MBit/s für ein Videosignal mit Standardauflösung bzw. bei über 1,4 GBit/s für ein hochauflösendes Signal liegt und mehrere Programme gleichzeitig auf einem Kanal übertragen werden sollen, müssen diese für die Verteilung über die genannten Übertragungswege komprimiert und vorbereitet werden.

Die Rohde und Schwarz GmbH & Co. KG entwickelt hierzu Systeme auf Basis von Standard-Serverkomponenten, die diese Audio- und Videosignale komprimieren und in ein sendefähiges Format konvertieren. Auf diesen Systemen arbeitet eine Videoverarbeitungskette, die für den Datenaustausch zwischen den einzelnen verarbeitenden Prozessen gemeinsame Speicherbereiche verwendet. Diese gemeinsamen Speicherbereiche enthalten neben den Nutzdaten auch Zusatzinformationen, die Rückschlüsse auf Fehlfunktionen innerhalb der Verarbeitungskette zulassen. Daraus ergibt sich die zentrale Frage dieser Diplomarbeit:

Auf welchem Wege und mithilfe von welcher Open-Source-Überwachungssoftware können diese Systeme, insbesondere die gemeinsamen Speicherbereiche der Verarbeitungskette, überwacht werden, um im Entwicklungsprozess den Datenfluss zwischen den einzelnen Prozessstufen analysieren zu können und somit Fehlfunktionen aufzudecken und einzugrenzen?

Zur Beantwortung dieser Frage wird zunächst geklärt, welche Aufgaben eine Monitoring-Software im Allgemeinen zu erfüllen hat und welche Funktionen hierzu notwendig sind, bevor auf den speziellen Anwendungsfall eingegangen werden kann und weiterhin analysiert werden kann, welche Besonderheiten sich aus diesem ergeben. Danach werden ausgewählte Monitoring-Systeme auf ihre Eignung in Bezug auf Überwachungs- und Analysemöglichkeiten für den Anwendungsfall hin untersucht. Abschließend wird ermittelt, wie ein geeignetes System effizient in die bestehende Infrastruktur integriert werden kann.

Das erste Kapitel gibt eine Einführung in die Thematik des Monitorings sowie einen kurzen historischen Abriss.

Innerhalb der Grundlagenbetrachtung im zweiten Kapitel stehen die Aufgaben eines Monitoring-Systems im Vordergrund. Dazu werden unterschiedliche Methoden der Datenerfassung klassifiziert sowie standardisierte und verbreitete Überwachungsmechanismen erläutert. Des Weiteren wird der Transportstrom der Videoverarbeitungskette thematisiert.

Im dritten Kapitel folgt eine Evaluierung der Videoverarbeitungskette als Zielsystem der Überwachung. Es enthält die aufgestellten Anforderungen an das System durch die Rohde und Schwarz GmbH & Co. KG, anhand derer ein Bewertungsmaßstab für ausgewählte Open-Source-Monitoring-Systeme entsteht. Mit Hilfe dessen ist es möglich, diese miteinander vergleichbar zu machen und ein passendes System auszuwählen.

Das optimale Monitoring-System wird in die bestehende Infrastruktur der Videoverarbeitungskette integriert. Das vierte Kapitel beschreibt dazu den allgemeinen Aufbau des gewählten Monitoring-Systems. Es beinhaltet eine Schilderung über Art und Weise der Integration, eingerichtete Konfigurationen sowie erforderliche Anpassungen zur Unterstützung der Datenflussanalyse.

Ob das Ziel, ein entsprechendes Monitoring-System anhand der aufgeführten Kriterien finden zu können und somit allen Anforderungen gerecht zu werden, erreicht wurde, überprüft abschließend das fünfte und damit letzte Kapitel.

Die innovativen Bestandteile dieser Arbeit konzentrieren sich auf die Kapitel drei und vier. Im dritten Kapitel sind hierzu die Erstellung der Anforderungen anhand der Vorgaben durch die Rohde und Schwarz GmbH & Co. KG zu nennen sowie die Erarbeitung des Bewertungsmaßstabes und der darauf basierenden Bewertungen der ausgewählten Überwachungssysteme. Da das gewählte Überwachungssystem als Software in die bestehende Videoverarbeitungskette integriert wird, sind im vierten Kapitel die Abschnitte des Entwurfes sowie der Implementierung Teil der Schöpfung. Der Fokus dieser Abschnitte liegt auf den zusätzlich benötigten Komponenten für die Überwachung, der speziellen Konfiguration sowie den notwendigen Anpassungen des Überwachungssystems.

1 Einführung

Wenn man das Monitoring definiert, so ist es die systematische Beobachtung und Überwachung eines Prozesses oder eines Systems mithilfe von technischen Mitteln. In wissenschaftlichen Bereichen wird die Überwachung von Systemen zur Gewinnung von Informationen und Daten herangezogen sowie zur Prüfung von Hypothesen. In der Technik ermöglicht das Monitoring durch Überwachung, Früherkennung und Ereignisreaktion ein Eingreifen in einen Prozessverlauf, wenn dieser definierte Grenzwerte annimmt. Die Ziele in diesen Bereichen sind die Sicherstellung eines dauerhaften und zuverlässigen Betriebs sowie die Optimierung des Prozessablaufes [1]. Der weitere Verlauf dieser Arbeit beschränkt sich auf das Überwachen von informationstechnischen Systemen und Umgebungen.

Das Überwachen eines Computersystems ist der Prozess des Beschaffens von Status- und Konfigurationsinformationen von unterschiedlichen Elementen des Systems [2] S. 111. Zu den weiteren Aufgaben gehören das Aufbereiten der gesammelten Rohinformation sowie das Erstellen von kompakten Darstellungen und Berichten, die von einem Beobachter, beispielsweise einem Systemadministrator, in Augenschein genommen werden, um auf mögliche Ereignisse reagieren zu können.

Dabei reichen die Anfänge dieser Systeme bis in die 1960er Jahre zurück. Zu diesem Zeitpunkt waren die eingesetzten Großrechner und Server zentral in großen Einrichtungen aufgestellt. Unter den Systemen gab es keine Verbindungen, wenn Daten ausgetauscht werden mussten, wurden diese per Magnetband ausgetauscht. Mit dem Aufkommen von Kleinrechnern in den 1970er Jahren zogen verteilte Systeme in die Industrie ein. So ersetzte AT&T 1977 sein Kontrollzentrum durch ein neues computergestütztes Netzwerk-Operationszentrum (NOC) mit nationalen und internationalen Statusanzeigen [5]. Verwaltet wurden die Systeme über herstellerspezifische Verwaltungskonsolen, welche typischerweise aus einer text-orientierten Anzeige für den aktuellen Status bestand und Änderungen in den Konfigurationsdateien des Servers erlaubten [2] S. 12. Die Zusammenarbeit der unterschiedlichen Verwaltungskonsolen war zu dieser Zeit nur beschränkt möglich. Mit der Revolution des Personal Computers in den 1980er Jahren, änderten sich die Verwaltungsaufgaben grundlegend [6]. Die Entwicklung von Netzwerktechnologien und Standards ermöglichte die Kommunikation von unterschiedlichen Rechnern untereinander. 1987 modernisierte AT&T das NOC und stattete es mit einer computergestützten Videoleinwand, bestehend aus 75 Monitoren, zur Anzeige der Netzwerkaktivitäten aus und Manager setzten Computersysteme ein, um detaillierte Netzwerkinformationen zu erhalten [5]. Ab diesem Zeitpunkt gehört die Verwaltung der eigenen Großrechner eines Unternehmens genauso zum täglichen

Geschäft wie die Verwaltung von mehreren hundert Personal Computern. Es entwickelten sich komplexe Computernetzwerke – Verwaltungsaufgaben wie die Synchronisation von Anwendungen, die Überwachung des Netzwerkes sowie die Störungsbehebung gestalteten sich zunehmend schwieriger. Ein offener Standard zur Verwaltung von Netzwerken wurde benötigt.

Gegen Ende der 80er Jahre wurden mehrere Projekte gegründet, die sich dieser Aufgabe annahmen. Letztendlich wurde das SGMP (Simple Gateway Management Protocol) [7] zum Standard SNMP (Simple Network Management Protocol) [8] weiterentwickelt. Der Grundstein hierfür wurde durch die Spezifikationen von SMI (Structure of Management Information) [9] sowie von MIB (Management Information Base) [10] gelegt. Die MIB bildet innerhalb von SNMP ein Standard Format zum Darstellen der zu verwaltenden Informationen. Sie reduziert so den Verwaltungsaufwand der Daten signifikant, denn nur mithilfe von standardisierten Formaten ist es möglich, Daten von verschiedenen Geräten miteinander zu vergleichen. Mithilfe des Agenten-Manager Modells, eine Nachahmung des Client-Server Modells, unterstützt SNMP eine einfache Weise zur Verwaltung und Überwachung von Netzwerken. Der Manager stellt dabei Verwaltungsfunktionen zur Verfügung, während die Agenten den Zugang zu den benötigten Informationen auf dem zu überwachenden Gerät herstellen. So erlaubt SNMP auch das Versenden von Aktualisierungsnachrichten an die Remote-Systeme.

Die DMTF (ehemals Desktop Management Task Force, heute Distributed Management Task Force) [12] machte es sich Mitte der 1990er Jahre zur Aufgabe, ein analoges Modell zu den SNMP MIBs zu entwickeln, dem Common Information Model (CIM). Dieses objektorientierte Modell ist in der Lage, benötigte Verwaltungsinformationen zu repräsentieren als auch Schnittstellen und Beziehungen unter den Komponenten darzustellen. 1998 verabschiedete die DMTF einen weiteren Standard, das Web-Based Enterprise Management (WBEM). Dieser enthält ein Transport Protokoll, ähnlich SNMP, für die webbasierte Verwaltung. Sowohl CIM als auch WBEM führen heute ein Schattendasein neben SNMP. SNMP hat sich als Standardprotokoll für das Netzwerkmanagement etabliert und liegt aktuell in Version 3 aus dem Jahre 2002 vor [2] S. 14. In der Praxis setzten sich in den letzten Jahren Management-Anwendungen durch, die von den aufkommenden Entwicklungen in der Computertechnik profitierten. Relationale Datenbanken werden für die Speicherung der Verwaltungsinformationen eingesetzt und die klassischen Desktop-Anwendungen der Verwaltungskonsolen werden durch browserbasierte Anwendungen ersetzt. Alle Verwaltungsoperationen werden in diesen Anwendungen durch Skripte bereitgestellt, die von einem Web Server ausgeführt werden. Benutzerfreundlichkeit, autonomes Computing und weitere Konzepte versuchen, die Managementsysteme weiter zu verbessern.

2 Grundlagen

2.1 Einordnung

In der Praxis erfordert die Verwaltung eines Computersystems die Überwachung verschiedener Daten. Zuständig dafür ist ein Monitoring-System, welches meist, wie in Abbildung 1, zu einem höheren Managementsystem gehört. Ein Managementsystem dient im Allgemeinen der Unterstützung eines Systemverwalters und soll dessen Effizienz maximieren. In Computersystemen wird je nach den verwalteten Komponenten vom Netzwerkmanagement bzw. vom Systemmanagement gesprochen [3].

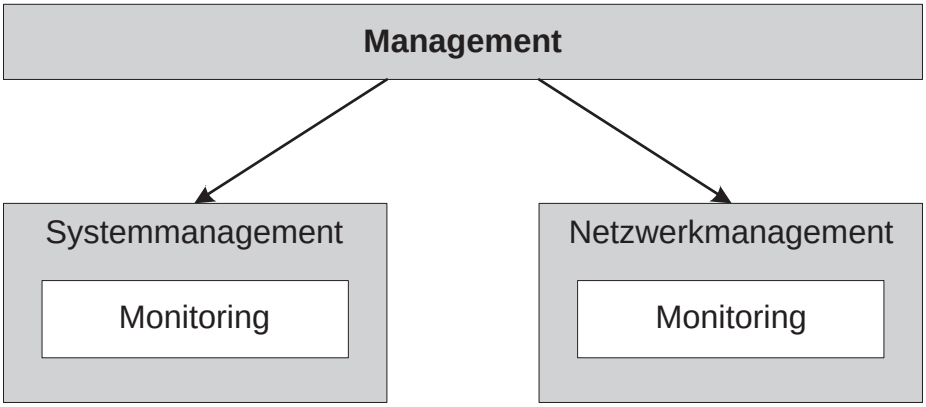

Abbildung 1 Kategorien IT-Management

Unter Netzwerkmanagement wird die Verwaltung, Überwachung und Sicherung von Rechnernetzen sowie von Telekommunikationsnetzen verstanden. Hierzu gehören unter anderem die Überwachung von Netzwerkkomponenten, die Messung der Netzwerkleistung sowie die zentrale Konfigurationsverwaltung der einzelnen Komponenten. Zur Bewältigung dieser Aufgaben werden rechnergestützte Netzmanagementsysteme eingesetzt, die den organisatorischen Aufwand auf ein Minimum reduzieren sollen. Das Systemmanagement beschäftigt sich mit Techniken, die zur Sicherstellung des fehlerfreien Arbeitens von verteilten Computersystemen benötigt werden. Ein System ist dabei eine aufgabenbezogene bzw. zweckgebundene Einheit von Elementen, die miteinander verbunden sind und in Wechselwirkung stehen [4]. Zu den Aufgaben des Systemmanagements gehören beispielsweise das Erfassen des Soft- und Hardwareinventars, die Benutzerverwaltung, die Softwareverteilung oder das Überprüfen der Serververfügbarkeit. In bestimmten Bereichen, wie der Telekommunikation, wird das Systemmanagement stark vom Netzwerkmanagement beeinflusst. In diesen speziellen Umfeldern ist eine klare Trennung von System- und Netzwerkmanagement nicht möglich. Das System-Monitoring ist also eine Disziplin des Systemmanagements zur Überwachung der eingesetzten Hardwarekomponenten und Softwareanwendungen eines spezifischen Systems sowie zur Aufbereitung der gewonnenen Informationen für die weitere Analyse.

2.2 Notwendigkeit einer Systemüberwachung

Computersysteme und Netzwerke sind für die moderne Industrie unverzichtbar. Unabhängig von der Größe des Unternehmens hängt dessen Erfolg auch vom fehlerfreien Arbeiten des unterstützenden Computersystems ab. Hinzu kommt der Einsatz von immer mehr werdenden Anwendungen, die die installierten Systeme komplexer werden lassen. Umso wichtiger wird es für IT-Dienstleister und Versorger, ihre Prozesse kontinuierlich zu verbessern, um so ihre Service-Level-Agreements (SLA) effizient gewährleisten zu können.

Das Service-Level-Agreement ist ein Vertrag zwischen Dienstleister und Auftraggeber, in dem die vom Dienstleister zu erbringende Leistungen festgehalten sind. Sie sollen dem Auftraggeber eine vertragliche Sicherheit bieten sowie transparente Kontrollmöglichkeiten ermöglichen und definieren – meist [13]

- die Verfügbarkeit und die Anbindung des Netzwerks,
- die Verfügbarkeit der Hardware,
- den Austausch defekter Hardware,
- die benötigte Zeit für einen Serverneustart,
- die Wartungsfenster,
- die Rechtsfolgen bei Nichteinhaltung der Vereinbarungen.

Die Verfügbarkeit eines Systems ist das Maß für die Erfüllung von bestimmten Anforderungen innerhalb eines Zeitraumes. Sie definiert sich:

$$V = \left(1 - \frac{\sum t_A}{t_B}\right) \cdot 100\ \%$$ (Formel 1)

Garantiert der Dienstleister die Verfügbarkeit seines Systems von 99,0 % während der Gesamtbetriebszeit t_B von einem Jahr, muss der Auftragnehmer mit einer Ausfallzeit t_A von 3,65 Tagen im Jahr rechnen. Je nach Anzahl der Systemnutzer ergibt sich so im Jahr eine potenzielle Schadenshöhe. Es gilt, die Kosten für Verfügbarkeit und die potenzielle Schadenshöhe in Balance zu halten. Das Beispiel aus Tabelle 1 veranschaulicht dies mit den Annahmen von 200 Mitarbeitern zu je 50 € pro Stunde einmal näher.

Tabelle 1 Schadenshöhe bei gegebener Verfügbarkeit

Verfügbarkeit	**99,0 %**	**99,99 %**
Ausfallzeit pro Jahr	3,65 d	0,0365 d
Kosten pro Stunde bei 200 Mitarbeiter mit 50 €/h	10.000,00 €	10.000,00 €
Potenzielle Schadenshöhe pro Jahr	876.000,00 €	8.760,00 €

Eine effiziente Überwachung der Systeme gehört für den IT-Dienstleister also zu den Kernaufgaben seines Managementprozesses, um effizient zu arbeiten, Ausfallzeiten zu verkürzen und die Verfügbarkeit garantieren zu können. Des Weiteren erlaubt ein System- und Netzwerk-Monitoring durch die Analyse von Trends ein vorausschauendes Planen von Systemressourcen, es erlaubt Präventivmaßnahmen und kann Schwachstellen in der IT-Infrastruktur aufdecken.

2.3 Aufgabenspektrum

2.3.1 FCAPS-Management

Da das System-Monitoring sehr eng mit dem Systemmanagement verknüpft ist, leiten sich die eigentlichen Überwachungsaufgaben aus den Aufgaben der Systemverwaltung ab. Die ISO bzw. die ITU-T definierte hierzu FCAPS, ein Managementmodell für Netzwerk- und Systemmanagement, in dem die primären Funktionalitäten festgehalten sind [14], [15]. Dazu gehören:

- Fehlermanagement,
- Konfigurationsmanagement,
- Abrechnungsmanagement,
- Performancemanagement,
- Sicherheitsmanagement.

Das Fehlermanagement beinhaltet Mittel zur Erkennung, Isolation und Behebung von auftretenden Fehlern. Im Störungsfall wird mithilfe der Root Cause Analysis (RCA) versucht, die gemeinsame Ursache von Fehlern festzustellen. Die Aufgabe des Konfigurationsmanagements ist es, Systeme zu identifizieren, zu kontrollieren, Daten von ihnen zu sammeln und Daten für sie bereitzustellen. Es hat das Ziel, Netzwerkverbindungen und Dienste vorzubereiten, zu initialisieren, zu starten, den kontinuierlichen Betrieb sicherzustellen und letztendlich auch zu beenden [15] S. 8. Das Abrechnungsmanagement beschäftigt sich mit der Datenerfassung, der Verwaltung und Überwachung von Ressourcen- und Kostenbegrenzungen sowie der Verwaltung von anfallenden Abrechnungsdaten. Das Performancemanagement ermöglicht die Auswertung des Ressourcenverbrauchs sowie die Bewertung der Effektivität von Kommunikationsprozessen. Typischerweise werden statistische Informationen von einer Komponente über einen längeren Zeitraum gesammelt, um Abweichungen vom Normalbetrieb, z. B. Lastspitzen, erkennen zu können. Zu den Aufgaben des Sicherheitsmanagements zählen das Erstellen, Löschen und Überwachen von Sicherheitsdiensten und Mechanismen, die Verwaltung von sicherheitsrelevanten Informationen sowie das Melden von relevanten Ereignissen.

In der Praxis hat sich die Management-Schrittkette aus Abbildung 2 etabliert, die alle Aspekte einer FCAPS Managementfunktion berücksichtigen soll. Sie besteht aus den Schritten der Feststellung, Überwachung, Analyse, gefolgt von Berichterstattung oder Rekonfiguration [2] S. 84.

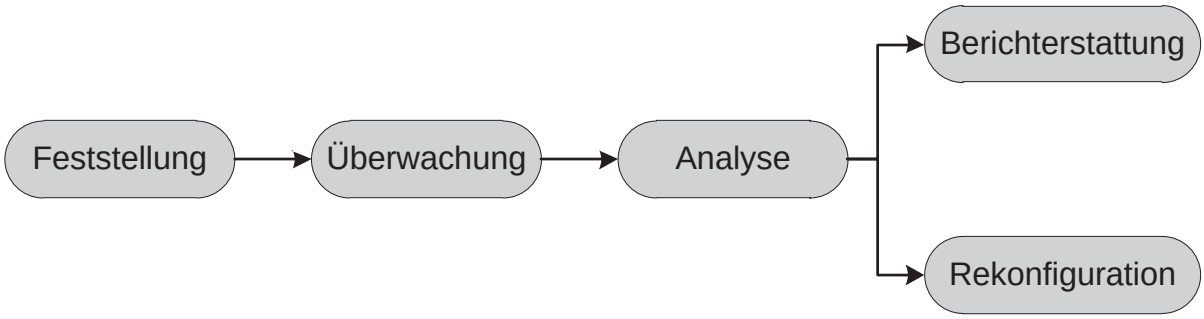

Abbildung 2 Management-Schrittkette

Im ersten Schritt muss festgestellt werden, aus welchen Elementen die Infrastruktur des zu überwachenden Systems bzw. Netzwerks besteht. Die gefundenen Elemente bilden das Inventar der IT-Infrastruktur und können sowohl physisch als auch virtuell, wie z. B. ein installiertes Softwarepaket, sein. Im nächsten Schritt wird das Inventar der Infrastruktur überwacht. Für jede Funktion im FCAPS Management werden andere Verwaltungsinformationen benötigt. Aufgabe der Überwachung ist es, die korrekten Informationen bereitzustellen. Im Analyse-Schritt werden die gewonnen Daten aufbereitet, um Informationen über den Systemzustand zu erhalten sowie angemessene Maßnahmen für aufkommende Probleme zu treffen. Die Analyse ist sehr stark von der FCAPS-Funktion abhängig. Während im Fehlermanagement die Analyse aus der Identifikation des Fehlers besteht, kombiniert sie im Performancemanagement Informationen von verschiedenen Quellen zu entsprechenden Metriken. Die Ergebnisse der Analyse werden anschließend an nachfolgende Instanzen weitergereicht. Dies kann z. B. ein Bericht über die aktuelle Systemperformance sein, der periodisch angelegt wird oder die Ergebnisse fließen in eine geänderte Konfiguration, beispielsweise um die Verfügbarkeit des Systems zu erhöhen.

Es zeigt sich, dass die einzelnen Schritte je nach FCAPS-Funktion unterschiedlich ausgeprägt sind. Die zu lösenden Aufgaben können mithilfe der Matrix aus Abbildung 3 für die jeweilige Systemdomäne genauer kategorisiert werden [2] S. 85. So liegt der Fokus bei einem Server, wie z. B. Web- oder Datenbankserver, auf der Sicherstellung des fehlerfreien Betriebs. Hierzu müssen System- und Anwendungsfehler erkannt und behoben, Leistungsengpässe ausgeglichen sowie Angriffe von außen abgewehrt werden. Abbildung 3 zeigt durch die schraffierten Flächen beispielhaft die zu lösenden Aufgaben eines entsprechenden Servers.

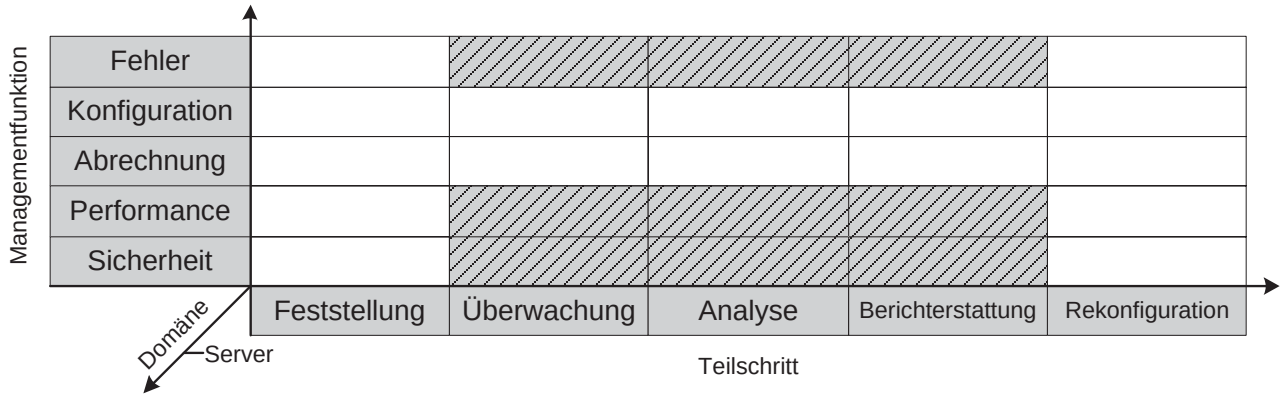

Abbildung 3 FCAPS-Management-Matrix

2.3.2 Überwachungsaufgaben

Für das Monitoring ergibt sich je nach Systemdomäne und Managementfunktion ein spezifisches Aufgabenspektrum.

Im Fehlermanagement wird eine ereignisgesteuerte Architektur eingesetzt, um ein System zu überwachen. Die Erkennung von Fehlern erfolgt durch Überwachungsmaßnahmen von Protokollen oder Entgegennahme von Fehlermeldungen. Die Isolation des Ursprungsfehlers ist gerade im Netzwerkbereich von besonderer Bedeutung, da hier der Ausfall von einzelnen Komponenten eine Kettenreaktion von Folgefehlern auslösen kann. Darüber hinaus werden die Systemkomponenten in periodischer Weise abgefragt, ob sie noch fehlerfrei arbeiten[1]. Dies kann durch einen unabhängigen Wächter erfolgen oder die Komponenten überwachen sich gegenseitig und melden einen Ausfall an das Verwaltungssystem.

Zu den typischen Funktionalitäten des Konfigurationsmanagements gehört:

- das Setzen von Konfigurationsparametern des Systems,
- die Namensauflösung von verwalteten Objekten und Objektgruppen,
- das Initialisieren und Stoppen von verwalteten Objekten,
- das Sammeln der aktuellen Konfigurationsparameter des Systems,
- die Entgegennahme von Meldungen über bedeutsame Zustandsänderungen,
- das Ändern der Systemkonfiguration.

Eine wichtige Aufgabe des Konfigurationsmanagement ist es, die Konfiguration der verwendeten Geräte in einem Unternehmen festzustellen. Eine Möglichkeit besteht darin, den Netzwerkverkehr zu überwachen, um Informationen über den Datenfluss zwischen den Geräten zu erhalten. So kann bspw. auf die laufende Anwendung auf dem Gerät

[1] Die Überprüfung der Komponenten erfolgt mit einer sogenannten Herzschlag-Nachricht. Wird auf diese Nachricht nicht geantwortet, muss von einem Ausfall der Komponente ausgegangen werden.

geschlossen werden. Des Weiteren werden die Parametereinstellungen einer Konfiguration überwacht, um Änderungen verfolgen und gegebenenfalls rückgängig machen zu können.

Während des Betriebes eines Systems ist es die Aufgabe des Performancemanagements, die Ursache für verminderte Leistung zu bestimmen und abzuschaffen sowie zukünftige Leistungsengpässe zu erkennen. Das kontinuierliche Überwachen der aktuellen Systemleistung ist dabei die wichtigste Maßnahme, um Leistungsschwächen aufzudecken. Die Leistungsüberwachung geschieht in Echtzeit oder in sich wiederholenden Intervallen. Des Weiteren nimmt das Performancemanagement Aufgaben zur Konfiguration des Systems und zur Verbesserung der Systemverfügbarkeit wahr.

Das Ziel des Sicherheitsmanagements ist es, eine ausreichend starke Verschlüsselung des Datenverkehres zu etablieren, effektiv vor Angriffen aus der Außenwelt zu schützen sowie die Zugänge zu schützenswerten Informationen zu überwachen. Um die Integrität eines Systems sicherzustellen, werden digitale Signaturen von allen Systemkomponenten erstellt. Innerhalb des Sicherheitsmanagements hat die Überwachung die Aufgabe, diese Signaturen zu erstellen und gegen verschiedene Listen, sogenannte Schwarze- oder Weiße-Listen, zu prüfen, um so unbeliebte Komponenten zu enttarnen. Hinzu kommt die Überwachung von Firewalls, Angriffserkennungssystemen und sogenannten Honigtöpfen. Ein Honigtopf ist ein speziell mit Schwachstellen aufgesetztes System, welches auf einen Angreifer besonders attraktiv wirken soll. Ziel ist es, den Angriff zu beobachten um effektive Gegenmaßnahmen entwickeln zu können.

Das Abrechnungsmanagement ermöglicht es Gebühren für die Ressourcennutzung einzuführen und identifiziert Kosten für die Belegung dieser Ressourcen. So verlangt jeder Internet Service Provider (ISP) eine Gebühr für den Zugang zum Internet. Meist erfolgt die Abrechnung volumenabhängig, daher ist die Erfassung des übertragenen Datenvolumens für die Abrechnung von existenzieller Bedeutung. Am Ende eines Nutzungszeitraumes muss für jeden Benutzer eines Systems eine Abrechnung der genutzten Ressourcen erstellt werden, um diese als Dienstleister in Rechnung stellen zu können. Das Abrechnungsmanagement muss Transaktionen und Anfragen einem bestimmten Nutzer zuordnen können. Hierfür ist die Überwachung von Ressourcen und des Netzwerkverkehres notwendig.

2.3.3 Überwachte Informationen

Um überhaupt eine Verwaltungsfunktion erfüllen zu können, müssen die wichtigen Informationen zuvor von allen Elementen des Systems gesammelt werden. Anschließend müssen diese Informationen einem Administrator oder einer Analysesoftware bereitgestellt werden, um diese weiterverarbeiten zu können. Dabei unterscheiden sich die benötigten Informationen je nach Verwaltungsfunktion sehr stark. Durch die unterschiedlichen Anforderungen der Verwaltungsfunktionen, lassen sich die benötigten Informationen in verschiedene logische Gruppen untergliedern [2] S. 111:

Statusinformationen: Sie werden für die Erfüllung fast aller Funktionen des FCAPS-Managements benötigt. Sie geben Auskunft darüber, ob ein Element des Systems ein- oder ausgeschalten ist und ob es wie erwartet arbeitet oder ausgefallen ist.

Konfigurationsinformationen: Die Konfiguration eines Elementes besteht aus allen Parametern, welche an die Bedürfnisse des Einsatzes angepasst werden können. Um sicherstellen zu können, dass Performanceverschlechterungen oder Systemausfälle nicht durch eine unzureichende Konfiguration verursacht werden, ist es notwendig, die Konfiguration eines Elementes zu überwachen.

Nutzungsstatistiken: Die Nutzungsinformationen eines Elementes besteht aus allen Attributen hinsichtlich ihres Durchsatz und die Anzahl der aktiven unterstützten Benutzer. Hierzu gehören Informationen wie die Anzahl der Pakete, der Benutzer oder der Sitzungen, welche auf dem System aktiv sind. Weitere Beispiele sind die Anzahl der genutzten Zwischenspeicher oder die Speicherauslastung.

Performancestatistiken: Zu den Performanceinformationen eines Elementes gehören die Verzögerungszeit sowie die Latenzzeit für die erfolgreiche Bearbeitung einer Anfrage an ein Element.

Fehlerinformationen: Zu diesen Informationen gehören alle Informationen über Fehler und fehlerhafte Operationen eines Elements. Sie können Fehlernummern sowie auch die Anzahl der Fehler, die in einer bestimmten Zeit auftraten, enthalten.

Informationen über die Netzstruktur: Diese Informationen werden im Feststellungsprozess gewonnen. Informationen über Änderungen an der Netzstruktur müssen als Teil des Überwachungsprozesses erhalten werden. In Systemen, in denen sich der Standort einiger Geräte ständig ändert, ist es essenziell diese Informationen zu erhalten. Das können z. B. Mobiltelefone oder mobile drahtlose Netzwerke sein.

2.4 Klassifizierung der Datenerfassung

Im ersten Schritt des Überwachungsprozesses müssen die benötigten Daten gesammelt werden. Von den einzelnen Geräten des Systems werden unterschiedliche Informationen gesammelt und in einer zentralen Datenbank abgespeichert. Dabei werden zwei Arten der Datenerfassung unterschieden. Während bei der passiven Überwachung das System anhand seiner gesendeten Daten beobachtet wird, werden bei der aktiven Überwachung eigene Anfragen an das System gestellt, um die gewünschten Informationen zu erhalten.

2.4.1 Passive Überwachung

Bei der passiven Überwachung werden die benötigten Informationen aus dem regulären Betrieb des Systems gesammelt, d. h. das Verwaltungssystem erhält die Informationen aus dem vorhandenen Datenstrom oder von einem auf dem System installierten Agenten, ohne es selbst zusätzlich durch Anfragen zu belasten. Der Agent sendet dabei die Informationen periodisch an den Verwaltungsserver. Dies ist ein wesentliches Merkmal der passiven Überwachung. Die einzige zusätzliche Last bzw. Arbeit ist diese, die zur Übertragung der genierten Informationen zum Verwaltungssystem benötigt wird.

2.4.1.1 Passive Anwendungsüberwachung

Die passive Überwachung kann in jedem System eingesetzt werden, welches Informationen für Verwaltungsaufgaben bereitstellen kann. Das können z. B. Protokolldateien sein, die von einer Anwendung auf dem System mit Informationen beschrieben werden. Die Informationen, die in die Protokolldatei geschrieben werden, sind von Anwendung zu Anwendung unterschiedlich. Einige Anwendungen schreiben Informationen über ihre internen Operationen in die Datei, aufgetretene Fehler oder den Anfragetyp, der bearbeitet wird.

Um die Informationen aus der Protokolldatei zu erhalten und diese an das Verwaltungssystem weiterzuleiten, werden Agenten auf dem System installiert, auf dem auch die Anwendung läuft. Die Agenten sind so in der Lage die Konfigurationsdateien zu lesen und zu ermitteln, unter welchem Pfad die Protokolldateien gespeichert sind. Die Agenten lesen periodisch die Protokolldateien, können sie bei Bedarf komprimieren und senden die Daten an das Managementsystem. Alternativ können die Protokolldateien durch einen Fernzugriff von einem anderen System ausgelesen werden. Die gesammelten Protokolldateien werden zur weiteren Verarbeitung auf dieses System über die aufgebaute Remote-Verbindung gesendet. Um die Protokolldateien nicht ins unendliche wachsen zu lassen, erlauben es viele Anwendungen, diese als Ringspeicher zu benutzen. Dabei werden die Informationen für einen bestimmten Zeitraum in der Datei gespeichert. Wenn

neue Informationen hinzukommen, werden die ältesten Einträge ersetzt und die Protokolldatei kann nur bis zu einer bestimmten Größe anwachsen.

Eine alternative Methode der passiven Anwendungsüberwachung ist der Einsatz eines Proxys. Dabei ist der Proxy eine Anwendung, die die gleichen Schnittstellen bietet, wie die zu überwachende Anwendung. Der Proxy wird zwischen dieser Anwendung und dem dazugehörigen Client-Netzwerk geschaltet. Auf diese Weise können die unterschiedlichsten Informationen gewonnen werden, wie z. B. die benötigte Zeit, die ein Server zur Beantwortung einer Anfrage benötigt, den Typ oder den Inhalt der Anfrage. Auf diese Weise lassen sich auch Informationen gewinnen, die mithilfe von Protokolldateien nur sehr umständlich erfasst werden können. Proxys sind für fast alle Anwendungen verfügbar, die als Server eingesetzt werden. Jedoch erhöht der Einsatz eines Proxys die Latenzzeit für die Bearbeitung einer Anfrage, erzeugt aber keine zusätzliche Last auf dem eigentlichen zu überwachendem Anwendungssystem. Die gewonnen Daten werden von den Proxys an das Verwaltungssystem gesendet oder speichern diese als lokale Protokolldatei ab, die vom Verwaltungssystem abgerufen werden kann.

2.4.1.2 Passive Überwachung von Rechnersystemen

Auf den verschiedensten Rechnersystem wie Servern, Personal Computern oder Laptops arbeiten meist unterschiedliche Betriebssysteme. Jedes der eingesetzten Betriebssysteme unterstützt verschiedene Methoden, um Informationen über das System zu erhalten. Eine übliche Methode ist die Bereitstellung von verschiedenen Betriebssystem-Kommandos, die das Sammeln von Schlüsselinformationen des Systems erlauben. Ein auf dem System laufender Agent kann so verschiedene statistische Informationen sammeln und an das Verwaltungssystem senden. Der Agent ruft dabei die bereitgestellten Kommandos, um die Informationen zu erhalten.

Es ergibt sich jedoch gerade bei der Überwachung von vielen Elementen ein potenzielles Problem beim Transport der Daten zum Verwaltungssystem. Wenn zu viele Agenten versuchen, ihre gesammelten Daten gleichzeitig an das Verwaltungssystem zu senden, wird dieses mit Anfragen überschwemmt. Das System kann überlastet werden und fällt im schlimmsten Fall aus. Um eine Überlastung des Verwaltungssystems zu verhindern, wird auf Zufallszahlen zurückgesetzt, um die Periodendauer für das Senden der Informationen zu erhalten. Dabei wird die fest definierte Periodendauer mit einem zufälligen Faktor eines definierten Wertebereiches verrechnet, sodass verschiedene Agenten ihre Daten nicht zur gleichen Zeit senden. Im Mittel werden die Daten innerhalb der Periodendauer versendet. Des Weiteren gibt es weitere anspruchsvolle Methoden, um die Last zu verteilen. Diese werden an dieser Stelle nicht weiter betrachtet.

2.4.1.3 Passive Netzwerküberwachung

Eine Methode, um ein Netzwerk passiv zu überwachen, ist der Einsatz von SNMP Traps (siehe Abschnitt 2.5.1.1). Ein Trap ist hier als eine Nachricht zu verstehen, die unaufgefordert von einem Agenten zum Verwaltungssystem gesendet wird. Die Netzwerkkomponenten können dabei so konfiguriert werden, dass sie bei verschiedenen Ereignissen diese Nachrichten erzeugen. Die geläufigste Methode der passiven Netzwerküberwachung ist das Beobachten der Datenpakete, die einen oder mehrere Punkte im Netzwerk passieren. Aus dem Fluss der Pakete und dem Wissen, dass die verschiedenen Netzwerkprotokolle ein bestimmtes Muster in ihrer Kommunikation aufweisen, können Informationen über die Anwendungen und Systeme, die mit dem Netzwerk verbunden sind, gewonnen werden. Um den Datenfluss untersuchen zu können, sind Kopien der im Netzwerk fließenden Pakete notwendig. Diese Kopien können auf unterschiedlichen Wegen erhalten werden [2] S. 118:

Port Mirroring: Mittels Port Mirroring wird der Netzwerkverkehr eines aktiven Anschlusses (Engl. Port) von einem Switch oder Router auf einen anderen Anschluss gespiegelt. Die Fehlersuche gestaltet sich beim Einsatz von Switches in einem Netzwerk schwierig, da die Pakete im Idealfall nur noch auf den Netzwerkpfaden unterwegs sind, die auch zum Ziel führen. Das Port Mirroring ist also notwendig um den Datenverkehr analysieren zu können. Hierzu wird an dem gespiegelten Port ein Rechner angeschlossen, der den Datenverkehr des aktiven Ports überwacht und gegebenenfalls analysiert. Das Port-Mirroring wird durch das SMON-Protokoll [16] standardisiert.

Netzwerk Taps: Netzwerk Taps[2] sind Geräte, die zwischen eine Netzwerkverbindung geschaltet werden, um Paketkopien des Datenverkehrs zu erhalten. Im engeren Sinne sind es Geräte, die auf das Port Mirroring spezialisiert sind. Je nach Herstellerspezifikationen werden ein oder mehrere Anschlüsse für die Überwachung des Datenverkehrs benötigt und zwei Anschlüsse, um die ursprüngliche Netzwerkverbindung herzustellen.

Broadcasting: In älteren drahtgebundenen Netzwerken werden die Datenpakete an alle sich im Netzwerk befindlichen Computer gesendet. Die eigentliche Empfangsstation eines Paketes verwirft alle anderen Pakete, die nicht für sie bestimmt sind. In diesen Netzwerken können einige Empfangsstationen so konfiguriert werden, dass sie Kopien von alle Datenpaketen entgegennehmen und diese zur weiteren Verarbeitung weiterleiten.

[2] Engl. Tap: Ein Hahn, Abzweigung oder Anzapfstelle

Moderne drahtlose Netzwerke sind ein Beispiel, in dem die Datenpakete an alle Empfänger gesendet werden.

IP Flow Information Export: IPFIX (Internet Protocol Flow Information Export) ist ein Standard, welches einer Netzwerkkomponente, wie z. B. einem Switch, erlaubt, bestimmte empfangene Datenpakete von zuvor festgelegten Maschinen oder Maschinengruppen zu kopieren und für die weitere Verarbeitung an ein Verwaltungssystem zu senden.

2.4.2 Aktive Überwachung

Bei der aktiven Überwachung wird zusätzliche Last durch Anfragen erzeugt, um Informationen aus dem Netzwerk zu erhalten. Auf diese Weise können Informationen gewonnen werden, die nur umständlich auf passivem Weg erhalten werden können.

2.4.2.1 Aktive Anwendungsüberwachung

Bei der aktiven Anwendungsüberwachung erzeugt die Überwachungssoftware eine künstliche Last auf der zu überwachenden Anwendung. Diese zusätzliche Last darf die normale Funktion des Systems nicht beeinflussen oder unterbrechen. So können Reaktionszeiten von Anwendungen, wie z. B. einem Webserver, oder die Betriebszeit gemessen werden. Durch Wiederholung mit einer angemessenen Periodendauer kann ein System kontinuierlich überwacht werden, ohne das System signifikant durch die künstliche Last zu beeinflussen.

Für viele Anwendungen ist es wünschenswert, das Verhalten des Systems bei Veränderung seines internen Status zu überwachen. Der Status einer Anwendung wird von den Informationen beschrieben, die die Anwendung in ihrer grundlegenden Funktion enthält und verwaltet. Bestimmte Operationen können den internen Status einer Anwendung ändern. Durch Erzeugen, Löschen und meist auch Aktualisieren von Informationen, die durch die Anwendung verwaltet werden, kann der Status der Anwendung geändert werden. Lese- und einige Aktualisierungsoperationen verändern hingegen den Status der Anwendung nicht. Um das Verhalten bei Statusänderung der Anwendung aktiv zu überwachen, existieren zwei übliche Wege.

Der erste Weg ist es, Dummy-Informationen zu erstellen. Deren einziger Zweck ist es, die künstliche Last auf dem zu prüfenden System zu erzeugen. Beispielsweise können über eine Verwaltungsanwendung im Bankwesen Dummy-Konten angelegt werden, die für Transferaktionen herangezogen werden. Diese Transaktionen führen dann zu Statusänderungen innerhalb der Anwendung. Die Statusänderungen betreffen jedoch nur die Dummy-Konten und haben keinen Effekt auf reale Konten der Bank.

Der zweite Weg besteht darin, zwei Aktionen zu nutzen um die Leistung des Systems zu bewerten. Eine Aktion macht dabei den Effekt der vorausgegangenen wieder rückgängig. Anknüpfend an das vorherige Beispiel im Bankwesen kann die erste Transaktion einen bestimmten Betrag von einem Konto auf ein anderes überweisen. Die zweite Transaktion überweist den gleichen Betrag wieder auf das erste Konto und macht den Effekt der ersten Transaktion rückgängig. Zusammengenommen verändern beide Transaktionen zusammen den Zustand der Anwendung nicht.

Im Allgemeinen lässt sich sagen, wenn die Kombination von verschiedenen Aktionen am Ende in keiner Statusänderung der Anwendung resultiert, dann kann das Überwachungssystem diese Aktionskombination benutzen, um verschiedene Informationen über die Anwendung zu gewinnen. Im Fehlerfall muss dafür gesorgt werden, dass die Anwendung in den Ursprungszustand zurückgesetzt werden kann.

2.4.2.2 Aktive Überwachung von Rechnersystemen

Die aktive Überwachung von Servern, tragbaren Geräten und Personal Computern kann mithilfe von installierten Agenten auf den Geräten oder ohne zusätzliche Software erfolgen. Wird ein Agent auf dem zu überwachenden Rechner eingesetzt, wartet dieser, im Gegensatz zur passiven Überwachung durch Agenten, auf Anfragen durch das Monitoring-System [2] S. 121. Die Aufgaben des Agenten sind also vergleichbar mit denen aus der passiven Überwachung, mit dem Unterschied, dass der Agent auf die Anfrage mit den geforderten Informationen antwortet, anstatt diese selbstständig zu senden.

Der Management-Server fragt dabei die installierten Agenten auf den Servern innerhalb des festgelegten Zeitraumes ab. Der Zeitraum und die Reihenfolge der Abfragen sind dabei so gestaltet, dass diese das Netzwerk nicht über ein akzeptables Maß hinaus belasten. Auf diese Weise kann das Monitoring-System nicht durch eine Überschwemmung von Anfragen, wie im Abschnitt 2.4.1.2 erläutert, überlastet werden. Die aktive Überwachung durch ein Management-System eignet sich daher besonders für die Überwachung von verschiedenen Geräten, die koordiniert werden müssen, um eine Überlastung des Monitoring-Systems zu verhindern.

Eine einfachere Herangehensweise ist die agentenlose Methode, bei der keine zusätzliche Software auf den Geräten benötigt wird. Typischerweise führt das Management-System ein Remote-Skript auf dem Gerät aus, um die benötigten Informationen auf dem Gerät zu sammeln. Der Schwerpunkt bei agentenlosen Systemen liegt in der Verwaltung der Anmeldedaten, welche dem Management-System erlaubt, sich auf dem Zielgerät einzuloggen, aber dabei unbefugten Benutzern keinen Zugang zu den Geräten und den Management-Informationen gewährt. Eine Lösung dieser Aufgabe ist die Verwendung eines üblichen Administrator-Zuganges zu allen Geräten und Servern in der zu

überwachenden Umgebung. Zum Problem wird diese Lösung, wenn die Zugangsdaten von einem Unbefugten kompromittiert wurden. Eine Alternative stellt die Speicherung der gerätespezifischen Passwörter in einem verwalteten Verzeichnis, welches über das Netzwerk erreichbar ist, dar. Das Management-System bekommt vom Verwaltungssystem des Verzeichnisses die Zugangsdaten für ein bestimmtes Gerät, sodass die benötigten Management-Informationen gesammelt werden können.

2.4.2.3 Aktive Netzwerküberwachung

Netzwerkgeräte können aktiv überwacht werden, indem die SNMP Agenten, die sich auf beinahe allen Netzwerkgeräten befinden, regelmäßig abgefragt werden. Diese Agenten stellen den Zugang zu allen Informationen bereit, welche auf dem Gerät im MIB-Format verfügbar sind [2] S. 122.

Zusätzlich zu SNMP können Programme wie „ping“ oder „traceroute“ benutzt werden, um Informationen über die Performance und Verfügbarkeit eines Netzwerkes zu erhalten. Diese Applikationen sind auf den meisten Computersystemen verfügbar und wurden als praktisches Mittel entwickelt, um zu überprüfen, ob ein anderes System im Netzwerk aktiv ist.

Das „ping“ Kommando erlaubt einem Rechner zu überprüfen, ob ein anderes Gerät im Netzwerk erreichbar ist. Das „traceroute“ Kommando versucht, alle Netzwerkknoten zu finden, die sich auf dem Weg zwischen dem Rechner, auf dem „traceroute“ ausgeführt wird, und einem zweiten entfernten Rechner befinden. Beide Programme liefern die geschätzte Latenzzeit der beiden Rechner im Netzwerk. Durch die Ausführung der Programme zwischen ausgewählten Punkten innerhalb eines Netzwerkes können hohe Latenzzeiten und verminderte Übertragungsraten zwischen Netzwerkknoten ermittelt werden. Der größte Vorteil dieser Programme ist, dass sie zur Ausführung keine administrativen Rechte benötigten. Jedoch sind sie in ihrer Ausführung ineffizient. Speziell „traceroute“ versendet mehrfach Nachrichten im Netzwerk und beobachtet diese auf Fehlschläge. Der übermäßige Gebrauch kann sich daher nachteilig auf die gesamte Netzwerkperformance auswirken.

2.5 Überwachungsmechanismen

In den vorhergehenden Abschnitten wurde schon mehrfach SNMP erwähnt, ohne näher auf das eigentliche Protokoll einzugehen. Neben SNMP werden in diesem Abschnitt weitere Protokolle und Methoden vorgestellt.

2.5.1 SNMP

Das bekannteste Protokoll aus dem Bereich Netzwerkmanagement ist SNMP. Es ermöglicht sowohl die Überwachung als auch die Konfiguration von Netzwerkkomponenten. Eingeführt wurde es 1988 und liegt mittlerweile in der dritten Version vor. In der ersten Version waren Sicherheitsfunktionen nur minimal vorgesehen gewesen. Erst mit der dritten Version wurden wichtige Sicherheitsmechanismen für die Praxis standardisiert.

2.5.1.1 Funktionsweise

SNMP arbeitet nach zwei wesentlichen Mechanismen. Zum einen ist es möglich, von einer entfernten Verwaltungsstation aus über eine definierte Schnittstelle Informationen zum Status von unterschiedlichen Geräten zu erfassen. Zum zweiten können die Geräte über eine einheitliche Schnittstelle konfiguriert werden. Der Datenblock, auch Protocol Data Unit (PDU) genannt, jedes SNMP-Pakets enthält die eigentlichen Anfragen und Nachrichten. SNMP Version 1 definiert für die vollständige Kommunikation fünf unterschiedliche PDU-Arten, die mit Einführung von SNMP Version 2 weiter ergänzt wurden [17] S. 138. Die Tabelle 2 beschreibt alle PDU-Arten näher.

Tabelle 2 SNMP PDU-Arten

PDU-Art	Beschreibung
get-request	Anforderung eines Datensatzes an einen SNMP Agenten
get-next-request	Wie get-request, jedoch wird der nachfolgende Datensatz gesendet
get-bulk-request	Anforderung von mehreren Datensätzen an einen Agenten
response	Antwort des Agenten auf SNMP Anfragen
set-request	Änderungen eines Datensatzes auf den mitgelieferten Wert
inform-request	Nachricht von einer Managementstation zu einer anderen
snmpV2-trap	Eine vom Agenten initiierte Nachricht an die Managementstation
report	Für zukünftige Erweiterungen

Die Kommunikation zwischen den Netzwerkkomponenten und der Managementstation kann uni- oder bidirektional verlaufen. Bei der unidirektionalen Kommunikation erfolgt das Versenden von Nachrichten von der überwachten Netzwerkkomponente zur

Managementstation. So ist es der Netzwerkkomponente möglich, ein außergewöhnliches Ereignis umgehend zu melden. Die bidirektionale Kommunikation wurde bereits im Abschnitt 2.4.2.3 erwähnt. Dabei stellt die Managementstation eine Anfrage an die überwachte Komponente und diese antwortet mit den geforderten Informationen. Auf diese Weise ist auch das Konfigurieren und Verwalten der Netzwerkkomponenten möglich [17] S. 70. Da die Hauptaufgabe eines Netzwerkes im Transport von Nutzdaten liegt und dieses minimal durch die Überwachung belastet werden soll, wird zur Kommunikation das verbindungslose User Datagram Protocol (UDP) eingesetzt. Auf den zu überwachenden Netzwerkkomponenten sind Agenten durch die Hersteller installiert, die auf Anfragen einer Managementstation antworten.

2.5.1.2 Management Information Base

Die Management Information Base (MIB) ist eine Datenbank zur Beschreibung der Informationen, die als sogenannte Managed Objects über ein Netzwerkprotokoll verwaltet werden können [18]. Jedes dieser per SNMP verwaltungsfähigen Objekte einer Netzwerkkomponente erhält eine eindeutige Identifikationsnummer, den Object Identifier (OID). Innerhalb der MIB sind die verschiedenen Objekte in einer hierarchischen Struktur angeordnet. Daraus ergibt sich auch die Struktur der OIDs. Diese bestehen aus einer Zahlenkette, getrennt durch einzelne Punkte wie z. B. .1.3.6.1.2.1. Dieser Object Identifier beschreibt den Pfad durch den globalen Baum, wie in Abbildung 4, zu den MIB-Modulen.

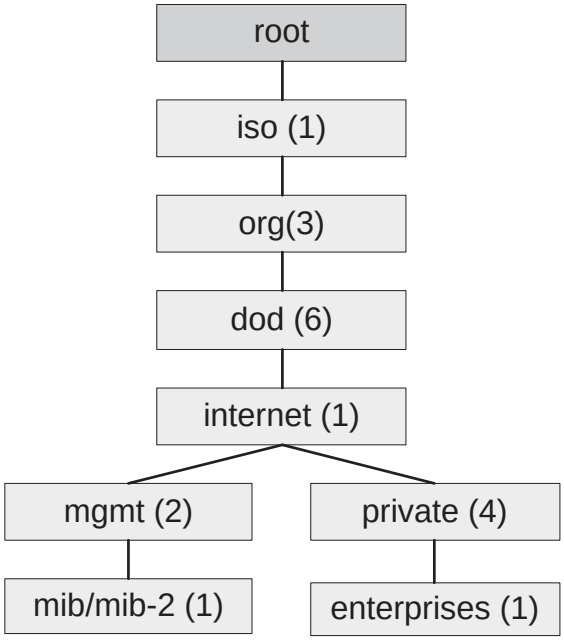

Abbildung 4 Globaler MIB-Baum

Zusätzlich sind den einzelnen Zahlen ASCII-Zeichenketten zugeordnet, die die Lesbarkeit erhöhen. Weitere Informationen sind im Request for Comments (RFC) 1155 [7] der Internet Engineering Task Force (IETF) zu finden.

Aus Abbildung 4 ist zu erkennen, dass sich die standardisierten MIBs nicht an der Wurzel der Baumstruktur befinden. Neben den standardisierten MIBs können unter dem Zweig „enterprises“ herstellerspezifische MIBs implementiert werden. Für MIBs existieren zwei verschiedene Standards, die MIB-I [8] und die MIB-II [18]. Dabei stellt die MIB-II eine Erweiterung zum Vorgänger dar. Sie enthält bis auf fehlerkorrigierte Objekte sämtliche Objekte der MIB-I in unveränderter Form und löste diese somit ab. Beide Standards definieren eine Reihe von allgemeinen Objekten, die zusätzlich durch eine Reihe weiterer MIBs ergänzt wurden. Ein Hersteller für SNMP-fähige Geräte muss die MIB-I bzw. die MIB-II implementieren. Alle anderen definierten MIBs sind optional. Die Abbildung 5 zeigt, welche Gruppen durch die MIB-II definiert sind.

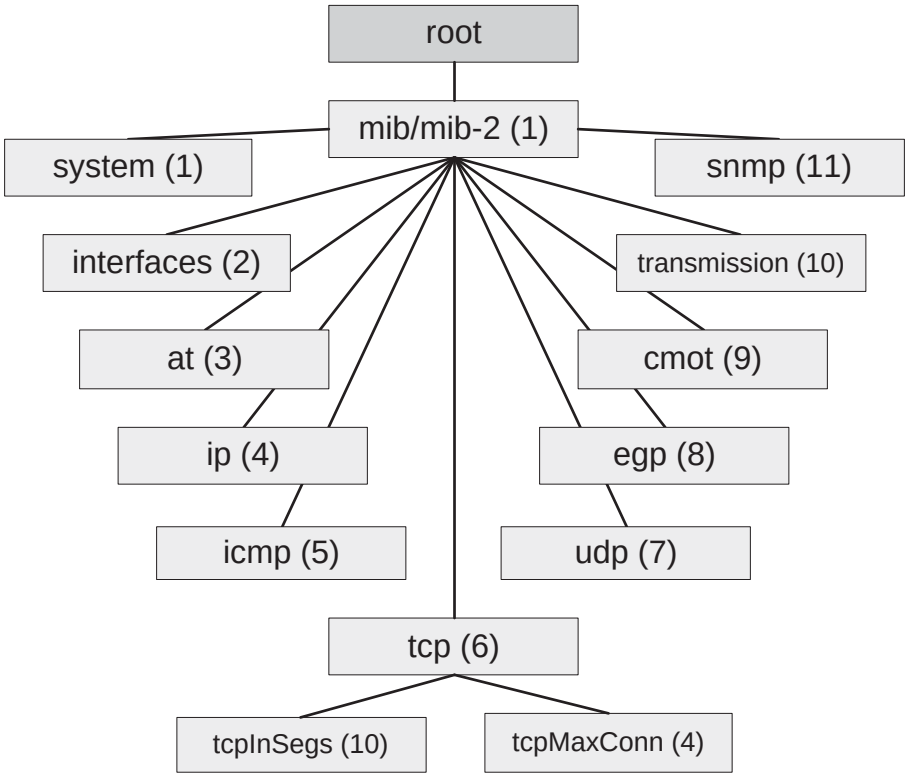

Abbildung 5 MIB-II Gruppen

Beispielsweise liefert das Objekt „tcpMaxConn“ mit der OID tcp.4 aus Abbildung 5 die maximale Anzahl paralleler TCP Verbindungen während das Objekt „tcpInSegs“ die Summe aller empfangenen TCP Pakete enthält.

2.5.1.3 Versionen

SNMP Version 1

Drei Standards bilden in Kombination miteinander SNMP Version 1. Die zu verwaltenden Objekte werden durch die Management Information Base [8] genau definiert. Zur Beschreibung der MIBs wird eine Definitionssprache mit Syntax [7] bestimmt sowie ein allgemeiner Protokollrahmen [19] aufgespannt. Das Protokoll beschreibt, über welche Kommunikationswege die Informationen zwischen Managementstation und Agenten ausgetauscht werden. Bereits erwähnt wurde, dass die Kommunikation über UPD verläuft. Dabei werden die SNMP Pakete über den Port 161 empfangen. Nachrichten mit der Trap-PDU werden jedoch vom Manager über den Port 162 empfangen.

Jede SNMP Nachricht besteht dabei aus einer Versionsnummer (Wert 1 für Version 1), den Community-Namen als Zeichenkette und dem Datenblock, welche die PDUs enthält. In Version 1 sind fünf PDU-Typen definiert: get-request, get-next-request, get-response, set-request und trap. Der Community-Name definiert eine zusammengehörige Gruppe von MIB-Zweigen und Managementstationen, um SNMP Pakete eindeutig dem Gesprächspartner der Community zuordnen zu können. Da der Name im Klartext mitgesendet wird, kann dieser nicht zur Authentifizierung genutzt werden. Aus diesem Grund wurden später SNMP Versionen mit wirksameren Sicherheitsmechanismen entwickelt.

SNMP Version 2

Im Laufe der Zeit entwickelten sich mehrere Varianten, die die wachsende Bedeutung von unterschiedlichen Sicherheitsaspekten berücksichtigen sollten. Letztendlich hat sich die SNMPv2c gegenüber den anderen Varianten durchgesetzt und breite Akzeptanz gefunden. Wenn von SNMPv2 gesprochen wird, ist damit meist SNMPv2c gemeint[3].

Auch das Regelwerk für SNMPv2 besteht aus einer Reihe von Dokumenten (siehe RFC 1901 bis 1907). Eine neue Definitionssprache, die SMIv2 [20], wird hierzu für alle SNMPv2 MIBs definiert. Die PDUs erfahren durch SNMPv2 kleinere Anpassungen, entsprechen aber im Wesentlichen denen aus Version 1. Insgesamt stehen in SNMPv2 acht unterschiedliche Typen von PDUs zur Verfügung (siehe Abschnitt 2.5.1.1). Neben der Nutzung von UDP als primärem Kommunikationsweg erlaubt SNMPv2 [21] weitere Wege wie z. B. das Datagram Delivery Protocol (DDP) im AppleTalk Protokollstapel. Mit

[3] In diesem Fall ist ebenfalls SNMPv2c gemeint, wenn von SNMPv2 die Rede ist.

SNMPv2 wird eine neue MIB [22] für SNMP Entitäten eingeführt, die zusammen mit SNMP verwendet werden kann.

SNMPv3

Mit der Entwicklung von SNMP Version 3 wurde das klare Ziele verfolgt, ein einheitliches Sicherheitskonzept einzuführen. Die Grundsteine hierfür stammen aus den unterschiedlichen Entwicklungen der einzelnen SNMPv2 Varianten.

Das RFC 3411 [23] beschreibt das neue Rahmenwerk von SNMPv3. In diesem stellen die SNMP Manger und Agenten die Implementierung einer Entität dar. Jede Entität besteht aus einer Funktionseinheit (einer verarbeitenden Einheit, ähnlich einem Prozessor) und meist mehreren Anwendungen wie z. B. Nachrichtenempfänger, Nachrichtensender oder Anfragenersteller. Die Funktionseinheit besteht aus mehreren Komponenten:

- Transport Subsystem
- Dispatcher (eine Art Nachrichtendisponent)
- Nachrichtenwandler (Message Processing Subsystem)
- Security Subsystem
- Zugriffskontrolle (nur für Agenten notwendig)

Da die Pakte bei SNMPv3 über verschiedene Protokolle eingehen können, müssen diese durch das Transport Subsystem übersetzt werden. Anschließend werden diese an den Dispatcher weitergereicht. Der Dispatcher, genauer gesagt der Message Dispatcher, extrahiert aus der Meldung die SNMP Version. Mithilfe dieser wird das Paket an den korrekte Nachrichtenwandler weitergeleitet. Von dieser Einheit bekommt der PDU-Dispatcher die enthaltene PDU zurück, um die richtige Anwendung ansprechen zu können. Der Nachrichtenwandler ist in der Lage, SNMP Anfragen und Nachrichten ineinander umzuwandeln. So kann er eine kodierte Nachricht in die korrekte PDU übersetzen. Je nach Sicherheitslevel muss der empfangene Datenblock noch entschlüsselt werden. Diese Aufgabe wird vom Security Subsystem übernommen. Aktuell sind zwei Security Modelle implementiert. Das User-based Security Modell unterstützt dabei die Überprüfung der Datenintegrität, Authentifizierung des Senders, Schutz gegen Verzögerung der Nachricht oder Wiederholung und Datenschlüsselung. Mithilfe des Transport Security Modell ist es möglich, SNMP über einen sicheren Transportkanal wie z. B. SSH oder TLS zu nutzen. Wird die Zugriffsschicht bei einem Agenten implementiert, entscheidet diese über die Berechtigungen der Anfrage. Weitere Informationen zu SNMP sind in den RFCs 3410 bis 3418 zu finden. Die Abbildung 6 veranschaulicht den Zusammenhang von Funktionseinheit und Anwendung noch einmal.

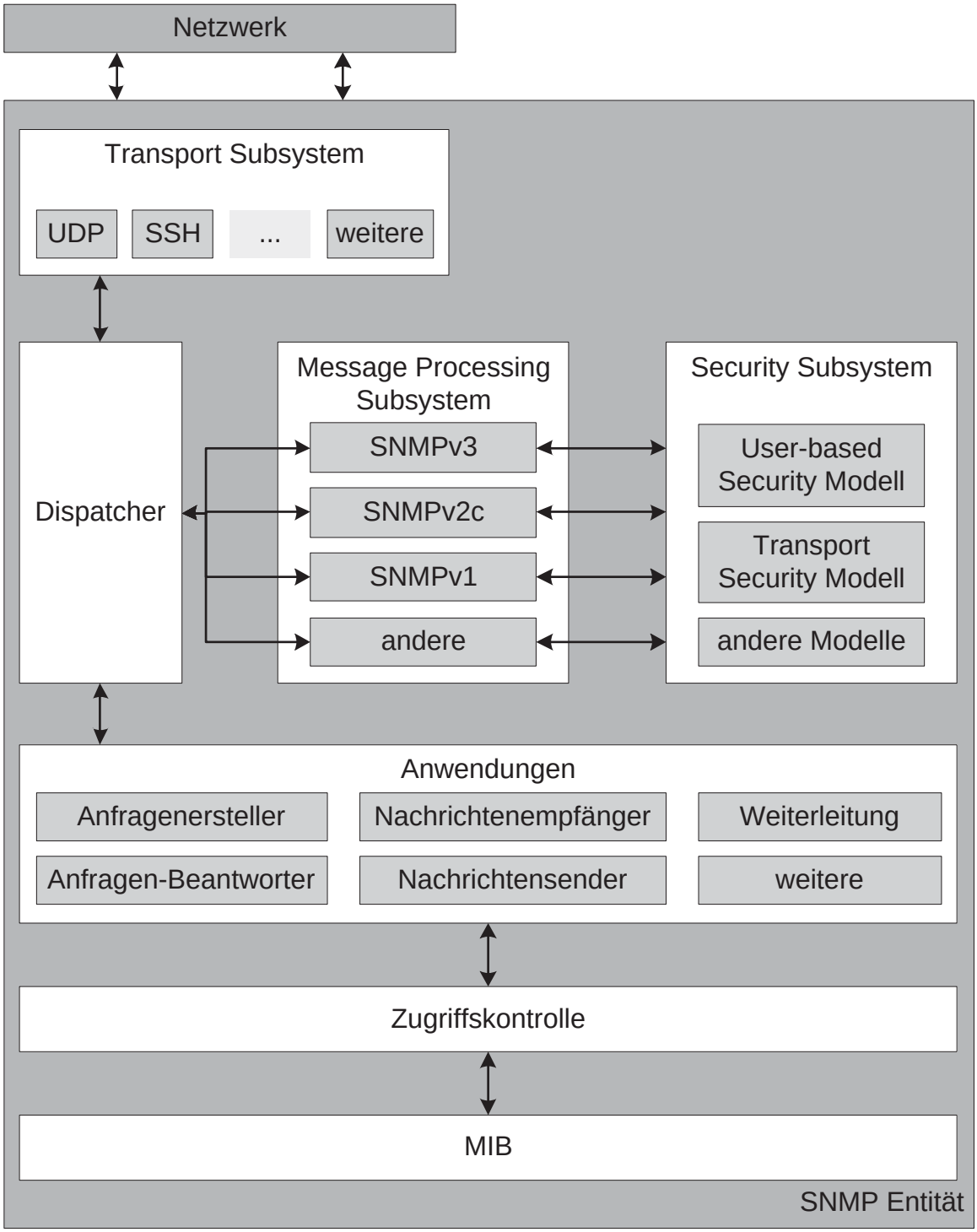

Abbildung 6 SNMPv3 Entität

2.5.2 ICMP

Das Internet Control Message Protocol (ICMP) ist Bestandteil der TCP/IP-Protokollfamilie und ist somit im Internet Protokoll IP eingebettet. Es hat zur Aufgabe, Fehler- und Diagnoseinformationen von einem Kommunikationspartner zu ermitteln. Für diese Aufgaben sind elf verschiedene Grundnachrichtentypen definiert [24]. Tabelle 3 gibt Aufschluss über die definierten Grundnachrichtentypen.

Tabelle 3 ICMP Nachrichtentypen

Typ	Name	Code	Erläuterung
0	Echo Antwort	0	Antwort auf eine Echo-Nachricht: muss den Datenbereich dieser unverändert wiederholen
3	Ziel nicht erreichbar	0	Netz nicht erreichbar: einem Gateway ist die Route zum Netzwerk des Paketzieles unbekannt
		1	Der Host ist für das Gateway nicht erreichbar
		2	Zielsystem unterstützt das verwendete Protokoll nicht
		3	Keine Paketzustellung auf angegebenen Port möglich
		4	Fragmentierung notwendig, da Paket maximale Länge überschreitet und DF-Flag (Don’t fragment) gesetzt ist
		5	Weiterleitung auf angegebener Route nicht möglich
4	Quelle unterdrücken	0	Paket verworfen, da der Empfangspuffer voll ist, eine Reduzierung der Datenrate ist notwendig
5	Umleitung	0	Umleitung für das gesamte Netzwerk, da eine effizientere Route bekannt ist
		1	Umleitung des Hosts über kürzeren Weg
		2	Umleitung für den Servicetyp und das Netzwerk
		3	Umleitung für den Servicetyp und den Host
8	Echo	0	Echo-Anfrage: wird durch Ping-Befehl verwendet
11	Zeit überschritten	0	Lebensdauer (Time to live) des Paketes abgelaufen
		1	Zeitlimit für Empfang der Paketfragmente überschritten
12	Parameterfehler	0	Paketkopf enthält syntaktische Fehler
13	Zeitstempel	0	Zeitstempel-Anfrage für Zeitmessungen
14	Zeitstempel Antwort	0	Antwort auf Zeitstempel-Anfrage
15	Informations-Anfrage	0	Veraltet. Anfrage auf Konfigurationsinformationen
16	Informations-	0	Veraltet. Antwort auf Informationsanfrage

	Antwort		

Neben den Grundtypen existiert eine Vielzahl weiterer Typen und Unterkategorien, die an dieser Stelle nicht weiter betrachtet werden[4]. Da die unterschiedlichen Nachrichtentypen andere Parameter erfordern, ist lediglich der ICMP Nachrichtenkopf bei allen Nachrichtentypen identisch. Der Paketkopf, auch Header genannt, enthält dabei den Nachrichtentyp, die Unterkategorie, den sogenannten Code und eine Prüfsumme über die gesamte Nachricht.

2.5.3 Protokollieren

Mithilfe des Protokollierens lassen sich alle internen Abläufe einer Anwendungen bzw. eines Systems nachvollziehen und so ein korrektes Verhalten nachweisen. Es ist daher eine wichtige Überwachungsmethode und bereits während des Entwicklungsprozesses unverzichtbar. Oftmals beschränkt sich die Protokollierung auf ein lokales System. Vom Betriebssystem bis hin zur Endanwendung werden die Nachrichten lokal auf dem System gespeichert. Syslog ist eine Anwendung auf Unix-basierten Betriebssystemen für die Ereignisprotokollierung, mit der auch die Protokollierung von Netzwerken möglich ist. Hierbei sendet eine Netzwerkkomponente eine Nachricht an den Syslog-Server. Die Kommunikation erfolgt jedoch unidirektional über UDP, d. h. die Reaktion auf ein Ereignis muss über einen anderen Weg erfolgen. Eine Syslog-Nachricht besteht aus einem Prioritäts-Selektor, einem Header und der eigentlichen Nachricht. Über den Selektor wird die Nachricht nach Schweregrad[5] und Herkunft klassifiziert[6]. Der Schweregrad kann dabei 8 Werte, von 0 für Notfall bis 7 für Debug-Informationen, annehmen. Der Header der Syslog-Nachricht enthält die IP-Adresse des Senders sowie den Zeitstempel bei Eingang der Nachricht. Jedoch gelten die definierten Formate [25] nicht als Internet-Standard, sondern stellen lediglich Empfehlungen dar. Im RFC 3195 [26] werden Ergänzungen beschrieben, die Syslog um einige Sicherheitsfunktionalitäten wie Verschlüsselung und Authentifizierung auf Basis des Transmission Control Protocols (TCP) erweitern [17] S. 157.

[4] Weitere Informationen sind auf der Website der Internet Assigned Numbers Authority (IANA) unter ICMP Parameter zu finden, siehe: http://www.iana.org/assignments/icmp-parameters/icmp-parameters.xml

[5] Engl.: severity

[6] Engl.: facility

2.6 Grundlagen des MPEG2-Transportstromes

Um ein Verständnis für die zu überwachenden Informationen zu erhalten, soll in diesem Abschnitt eine Einführung in den MPEG2-Datenstrom erfolgen. Dieser bildet die Grundlage der weiteren Verarbeitung. Details zur Videocodierung werden nicht näher betrachtet, da diese sich lediglich auf die Bildqualität und die Datenrate auswirken. Im Fokus steht jedoch der Aufbau des Datenstromes.

Unmittelbar nach der Komprimierung, dem Encodieren, stehen die allgemeinen Datenströme, die Audio- und Videosignale als Elementarströme (ES) für die weitere Verarbeitung zur Verfügung. Die Elementarströme werden in Pakete variabler Länge zerlegt. Dieser in Pakete untergliederte Elementarstrom wird Packetized Elementary Stream (PES) genannt [19] S. 37. Die Elementarstrompakete können bis zu 64 kByte groß sein und beginnen mit einem PES-Header, gefolgt vom Nutzlastanteil, die sogenannte Payload. Wie in Abbildung 7 zu sehen, beginnt der PES-Header mit einem 3 Byte großen Start Code Präfix mit dem Wert 0x00 00 01[7]. Der Start Code kennzeichnet ein PES-Paket als solches. Das vierte Byte im PES-Header ist die Stream ID, die die Art des Elementarstromes beschreibt, z. B. ein Video-, Audio- oder Datenstrom. Über den 2 Byte großen Bereich für die Paketlänge lassen sich die bis 64 kByte Speicher ansprechen. Videopakete können die Paketlänge von 64 kByte auch überschreiten. Hierfür besitzt diese den Wert 0x00 00.

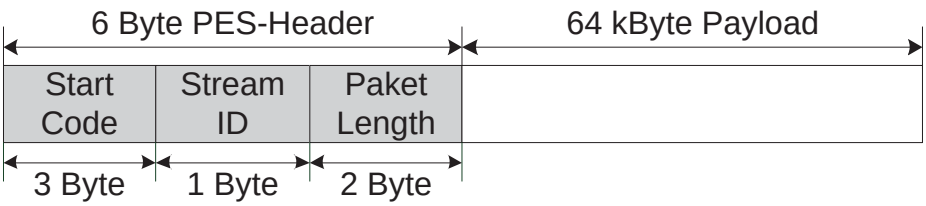

Abbildung 7 PES-Header

Der 6 Byte lange PES-Header kann durch einen optionalen Header im Nutzlastanteil erweitert werden. Dieser optionale Header ist abhängig vom Elementarstrom selbst und wird durch 11 Schalter[8] gesteuert. Diese Schalter kennzeichnen, welche Inhalte in den optionalen Feldern wirklich enthalten sind. Das Feld PES-Header Data Length gibt Aufschluss über die Gesamtlänge des PES-Headers. Die optionalen Felder enthalten unter anderem den Presentation Timestamp (PTS) sowie den Decoding Timestamp

[7] 0x ist das Präfix in C/C++ für einen hexadezimalen Wert

[8] Engl.: Flags

(DTS), die für die Synchronisation von Video- und Audiosignalen benötigt werden. Den Aufbau des optionalen PES-Headers veranschaulicht Abbildung 8 noch einmal näher.

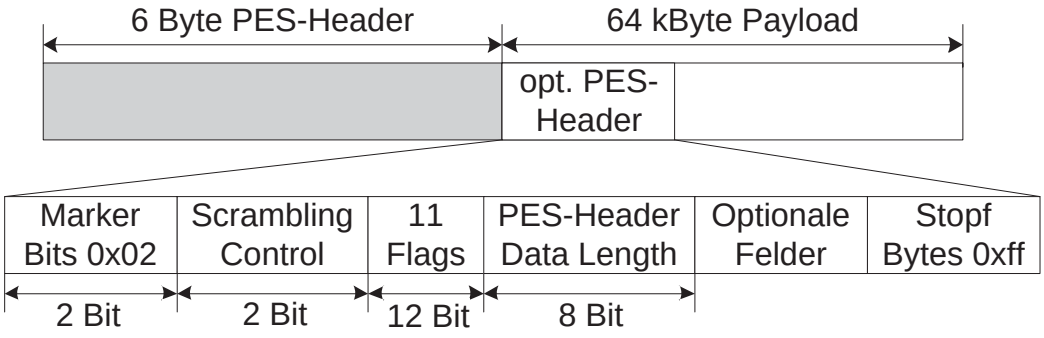

Abbildung 8 Optionaler PES-Header

Um bei MPEG2 ein gemeinsames Datensignal aus mehreren Programmen über Satellit, Kabel oder ein anderes Netzwerk übertragen zu können, werden die langen PES-Pakete, wie in Abbildung 9, in kleinere Transportstrompaket mit einer konstanten Länge zerlegt [19] S. 40. Die Transportstrompakete bestehen aus einem 4 Byte großem Header und aus einem184 Byte großen Teil des PES-Paketes.

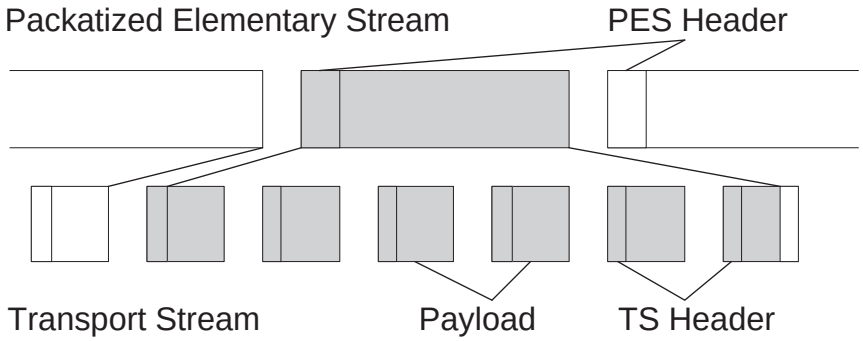

Abbildung 9 MPEG2 Transportstrompaket

Das erste Byte des Headers eines jeden Transportstrompakets ist das sogenannte Sync-Byte mit dem Wert 0x47. Es dient, zusammen mit der Information, dass es in einem Abstand von 188 Bytes im Datenstrom enthalten ist, zur Synchronisation auf den Transportstrom. Alle Datenströme eines Programmes können nun ineinander verschachtelt werden und bilden somit den eigentlichen Transportstrom (TS). Des Weiteren enthält der Header den 13 Bit langen Paket Identifier (PID). Neben den beschriebenen enthält der TS-Header weitere Informationen, die aber an dieser Stelle nicht weiter betrachtet werden, da sie nur unwesentlich zum Grundverständnis beitragen. Über die PID und weitere Tabellen ist es möglich, die Payload des Pakets zu identifizieren und einem Elementarstrom zuzuordnen. Abbildung 10 veranschaulicht den beschriebenen Aufbau.

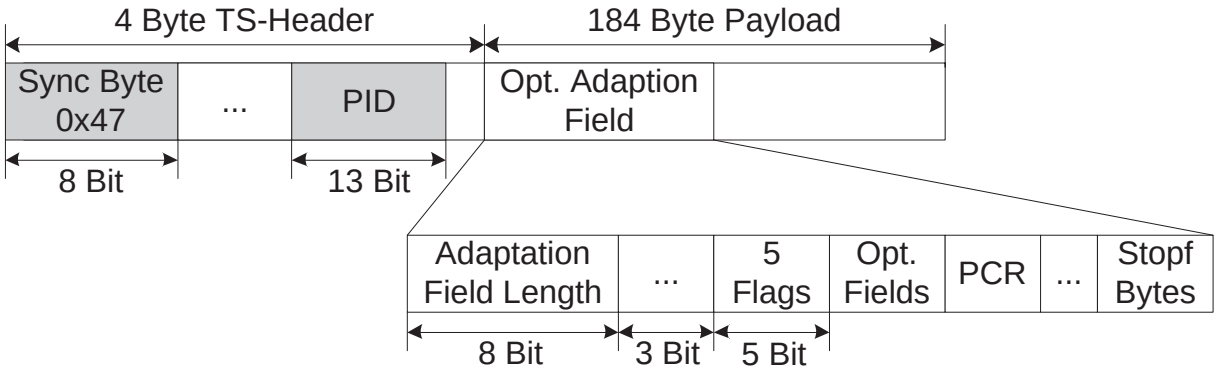

Abbildung 10 TS-Header

Diese Tabellen werden Programm spezifische Informationen (PSI) genannt und werden in einem definierten Abstand im Transportstrom übertragen. Zu diesen Tabellen gehört die Programm Association Table (PAT), welche die Anzahl der Programme im Transportstrom enthält. Sie wird alle 500ms wiederholt und existiert pro Transportstrom genau einmal. Transportstrompakete, die diese Tabelle enthalten, besitzen die PID null. Im Nutzlastanteil dieser Pakete befindet sich eine Liste mit PIDs, die auf weitere Tabellen, die sogenannten Program Map Tables (PMT), verweisen. Die enthaltenen PIDs der PAT stellen gewissermaßen Zeiger auf die PMTs dar. Die Tabellen der PMTs enthalten die PIDs für alle Programm spezifischen Elementarströme, also für alle Audio-, Video- und Datenströme eines Programms. Transportstrompakete, welche die PMTs enthalten, sind also spezielle Pakete, deren PIDs in der PAT festgehalten sind. Wird beispielsweise durch den Benutzer das Programm 1 gewählt, erhält man über die PAT die PID der dazugehörigen PMT. Über die PMT können wiederum die PIDs der Audio- und Videoströme erhalten werden. Ein Programm kann auch mehrere Audio- und Videoströme enthalten, letztendlich legt sich der Benutzer auf genau einen Audio- und Videostrom fest. Dieser Zusammenhang wird noch einmal durch Abbildung 11 veranschaulicht.

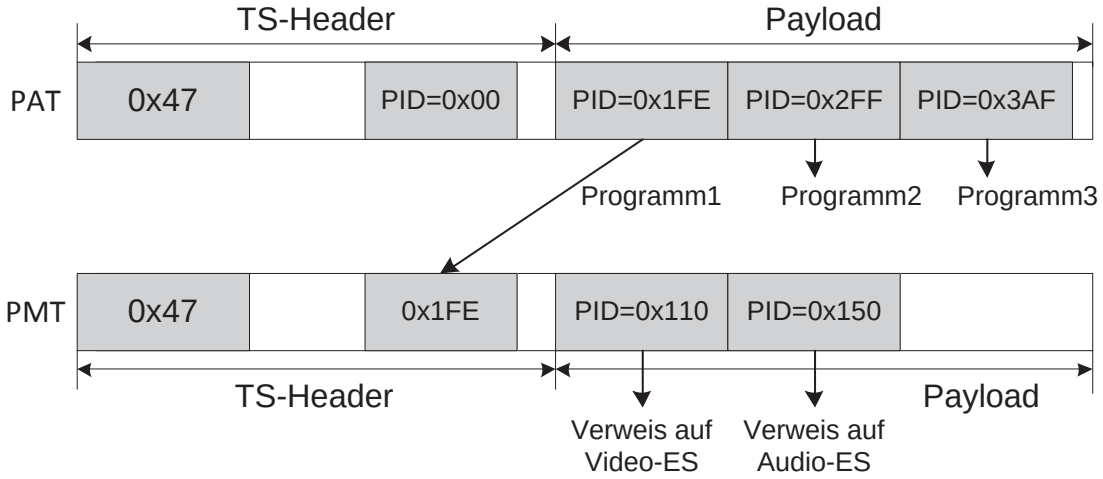

Abbildung 11 PAT und PMT

Auf diese Weise erhält der Demultiplexer die PIDs der gewählten Transportstrompakete der einzelnen Video- und Audioströme eines Programms. Nach dem Demultiplex-Vorgang stehen somit wieder die ES-Pakete für die weitere Verarbeitung durch den Decoder bereit. Der Decoder muss sich zuvor auf den Takt des Encoders synchronisieren. Hierzu wird im Transportstrom innerhalb des Adaption Fields die Program Clock Reference (PCR) mitgesendet (siehe Abbildung 10), welche als Referenzinformation zur Synchronisation dient. In der PMT wird festgehalten, in welchem Transportstrompaket sich der PCR-Wert befindet. Dieser ist eine Kopie des System Time Clocks (STC) des Encoders. Der Decoder vergleicht seine Systemzeit mit dem empfangenen PCR-Wert ab und regelt bei einer Abweichung seine STC nach. Pro Programm werden die PCR-Werte im Minimum aller 40ms übertragen [19] S. 54.

Nach erfolgtem Abgleich der Systemuhren von Encoder und Decoder können die Audio- und Videoströme synchron decodiert werden. Hierzu werden zusätzliche Zeitinformationen in die PES-Header der Audio- und Videoströme hinterlegt. Diese, von der Systemuhr abgeleiteten Zeitinformationen, werden Presentation Timestamps (PTS) genannt und in einem maximalen Abstand von 700ms wiederholt. Um Speicher im Decoder sparen zu können, kann die Übertragungsreihenfolge der komprimierten Bilder eine andere sein als die Aufzeichnungsreihenfolge. Um die Bildreihenfolge des Originals wieder herzustellen, werden zusätzliche Zeitmarken, die sogenannten Decoding Timestamps (DTS), im PES-Header übertragen. Dieser 33Bit Wert ist ebenfalls von der Systemuhr abgeleitet und enthält die höherwertigen Bits des PCR-Wertes.

3 Evaluierung

3.1 Einführung in die Videoverarbeitungskette

Die Aufgabe der Videoverarbeitungskette besteht darin, die eingehenden Datenströme aus allgemeinen Daten, Video- und Audiosignalen in ein gewünschtes Format zu transcodieren, zu multiplexen und in ein sendefähiges Format zu überführen. In diesem Abschnitt soll der Aufbau und Prozess der Verarbeitung näher betrachtet werden.

3.1.1 Aufbau der Videoprozesskette

Der Aufbau der Prozesskette unterscheidet sich, je nachdem welche Eingangsdaten verarbeitet werden. Dabei ist die Verarbeitung von bereits komprimierten Transportströmen als auch von nicht-komprimierten Videosignalen möglich.

In Abbildung 12 ist der Aufbau zur Verarbeitung eines eingehenden MPEG2-Transportstromes dargestellt. Die Daten werden zwischen den Prozessstufen (in Abbildung 12 grau hinterlegt) mittels einem gemeinsamen Speicher (siehe Abschnitt 3.1.2) ausgetauscht. Hinter den Prozessstufen stehen je nach Konfiguration unterschiedliche Anwendungen, die auf einem Server ausgeführt werden. Als Quelle des Transportstromes kann beispielsweise eine Datei im Transportstromformat, eine RTP-Netzwerkquelle[9] oder ein asynchrones serielles Interface (ASI) dienen, über welches der Transportstrom beispielsweise im digitalen Kabelnetz übertragen wird. Um den Transportstrom zu verarbeiten, muss dieser zuvor demultiplext und in einen dekomprimierten Elementarstrom umgewandelt werden.

Der eingehende Transportstrom wird hierzu durch den Demultiplexer in die verfügbaren Elementarstrompakete zerlegt. Neben Audio- und Videoströmen können im Transportstrom auch allgemeine Datenströme z. B. für Videotext oder Untertitel enthalten sein (in der Abbildung unten nicht dargestellt). Im Anschluss an das Demultiplexen erfolgt das Decodieren des Videostromes in einen dekomprimierten Strom im YCbCr-Farbformat[10] bzw. des Audiostromes in das PCM-Audioformat[11]. Im Vorverarbeitungs-Schritt wird die Auflösung des Videosignals an die Vorgaben des Ausgabeformates angepasst. Anschließend erfolgt das Encodieren des Audio- und Videosignales in das

[9] Real-Time Transport Protocol (RTP) zur Übertragung von audiovisuellen Datenströmen

[10] Farbmodell im digitalen Fernsehbereich, welches die Farbinformation in Helligkeit (Y) und zwei Farbigkeits-Signale (Cb und Cr) aufteilt.

[11] Puls-Code-Modulation (PCM), zur digitalen Darstellung von analogen Audiosignalen.

gewählte Zielformat sowie das Multiplexen der Elementarstrompakete eines Programms in den ausgehenden Transportstrom.

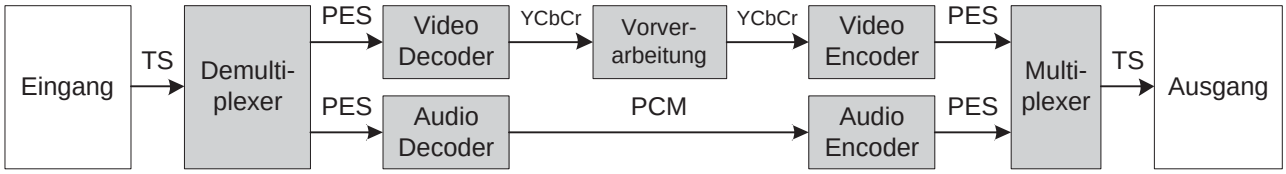

Abbildung 12 Aufbau der Prozesskette

Bei der Verarbeitung von nicht-komprimierten Datenströmen entfallen die Prozessstufen des Demultiplexens und Decodierens. Die Datenströme werden hierzu über ein serielles digitales Interface (SDI), welches im Bereich der professionellen Fernsehtechnik eingesetzt wird, empfangen und direkt den verarbeitenden Prozessstufen des Audioencoders bzw. der Videovorverarbeitung zugeführt. Das SDI-Signal enthält dabei die unkomprimierten Bildinformationen im YCrCb-Farbformat sowie bis zu 16 unkomprimierte Audiokanäle.

3.1.2 Interprozesskommunikation der Videoprozesse

Um die Daten zwischen den einzelnen Verarbeitungsprozessen auszutauschen, werden für die Interprozesskommunikation (IPC) gemeinsame Speicherbereiche (Engl. Shared Memory[12]) genutzt, die von mehreren Prozessen gelesen und von einem Prozess beschrieben werden. So werden die Daten effizient zwischen den einzelnen Prozessen ausgetauscht, ohne zusätzliche Kopiervorgänge zu benötigen.

Die Verwendung von Shared Memories eröffnet weitere Vorteile, die ebenfalls für den Einsatz sprechen. Dazu gehören das einfachere Debugging der einzelnen Komponenten, da jeder Prozessschritt entsprechende Ausgaben erzeugt, die wiederum analysiert werden können, sowie eine hohe Erweiterbarkeit mit neuen Komponenten. Über einen hinzukommenden Mechanismus muss jedoch der lesende und schreibende Zugriff auf diesen Speicher koordiniert werden, um die Konsistenz der Daten nicht zu gefährden.

3.1.2.1 Speicherstruktur

Allen Shared Memories liegt eine definierte Speicherstruktur zu Grunde. Sie bestehen aus einem 44 Byte langen globalen Header sowie aus einem oder mehreren Slots mit den dazugehörigen Daten innerhalb der Payload sowie den Slot Header. Die Größe des

[12] Da es sich bei einem Shared Memory um eine wesentliche Computertechnologie handelt, wird im weiteren Verlauf der englische Begriff Shared Memory verwendet.

Headers wird in den ersten 4 Bytes gespeichert. Des Weiteren enthält der globale Header eine einmalige ID des Shared Memories, über die ein schreibender Prozess mehrere Shared Memories verwalten kann. Mithilfe des virtuellen Zeitstempels können die enthaltenen Nutzdaten, in Kombination mit einem gültigen Zeitstempel, auf ihre Aktualität überprüft werden. Der gültige Zeitstempel enthält dabei einen Wert des virtuellen Zeitstempels, bis zu welchem die Nutzdaten gültig sind.

Die gespeicherten Nutzdaten werden durch einen entsprechenden Slot näher beschrieben. Dieser enthält die grundlegenden Beschreibung der Daten und gibt beispielsweise Auskunft darüber, an welcher Adresse des Shared Memories die Daten zu finden sind und ob diese gültig sind. Jeder Shared Memory kann mehrere Slots mit den dazugehörigen Daten enthalten. Die grundlegende Struktur eines Shared Memories visualisiert Abbildung 13 noch einmal.

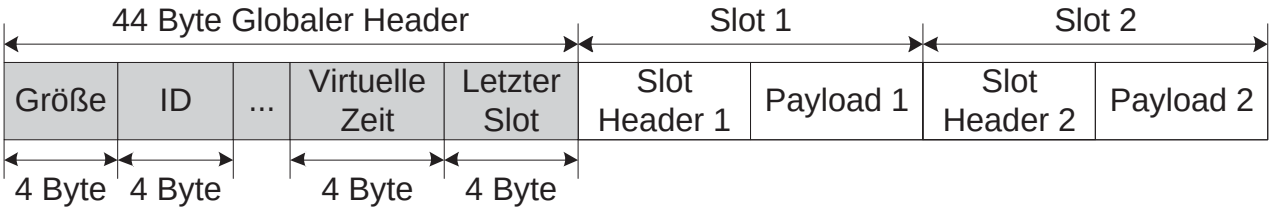

Abbildung 13 Speicherstruktur eines Shared Memory

Ein Slot besitzt einen ähnlichen Aufbau und besteht aus einem 20 Byte großen Block sowie einer Liste von zusätzlich angehangenen Metadaten. Die Größe beschreibt dabei die Gesamtlänge eines Slots und ist mindestens 20 Byte lang, für den Fall, dass keine Metadaten enthalten sind.

Das Typen-Feld gibt Auskunft über das Format der Nutzdaten. Dabei stehen Typenzeichen für verschiedene Datencontainer sowie nicht-komprimierte und komprimierte Video- und Audiosignale, beispielsweise AC-3 als Kodierungsverfahren des Dolby Digital Tonsystems, zur Verfügung. Über das Offset-Feld kann der Beginn der Nutzdaten in Bezug auf das erste Byte des Shared Memories ermittelt werden. Die 4 Byte lange Längenbeschreibung gibt Aufschluss über die Gesamtlänge der Nutzdaten.

Die Konsistenz dieser Daten kann mithilfe des gültigen Zeitstempels überprüft werden. Ist der virtuelle Zeitstempel gleich oder größer als der gültige Zeitstempel, können die enthaltenen Daten teilweise oder vollständig überschrieben worden sein. Abbildung 14 stellt den Aufbau eines Slots noch einmal grafisch dar.

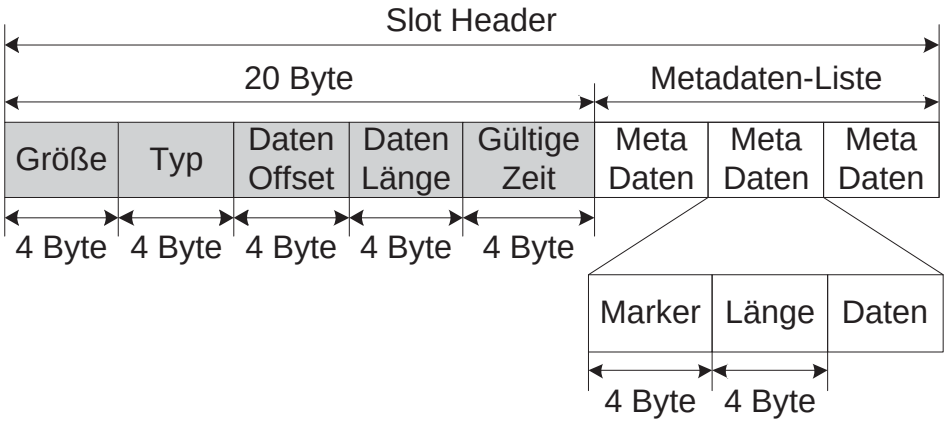

Abbildung 14 Struktur eines Shared Memory Slots

Die Liste der Metadaten enthält häufige sowie spezielle Zusatzinformationen der Nutzdaten. Die Metadaten bestehen aus einem Marker[13], der den Typ der Daten spezifiziert, einer 4 Byte langen Längenbeschreibung der Metadaten sowie den Metadaten selbst. Zu den häufigen Metadaten zählen unter anderem verschiedene Zeitstempel wie zum Beispiel DTS (siehe Abschnitt 2.6) sowie die Verarbeitungsverzögerung durch die Prozesskette. Der PTS ist dabei der einzige Zeitstempel, der immer vorhanden ist.

Je nach Format der Nutzdaten können weitere spezielle Metadaten enthalten sein. Während bei einem nicht-komprimiertem Videosignal Informationen zur Videoauflösung und dem Anzeigeseitenverhältnis zur Verfügung stehen, können die Metadaten bei einem nicht-komprimiertem Audiosignal Informationen zur Abtastrate sowie die Anzahl der Kanäle enthalten.

3.1.2.2 Zugriffssynchronisation

Um die Zugriffe der einzelnen Prozesse auf den Shared Memory zu koordinieren, wird ein proprietäres Protokoll auf Anwendungsebene eingesetzt. Dieses Protokoll nutzt für Menschen lesbare Nachrichten, die unter den Prozessen mithilfe des Transportprotokolls ausgetauscht werden.

Die Kommunikation unter den Prozessen ist dabei dem Entwurfsmuster eines Beobachters[14] nachempfunden. Dabei meldet sich der Beobachter, in diesem Fall der lesende Prozess, bei dem schreibenden Prozess an, um über Änderungen des beobachteten Objektes informiert zu werden [20]. Der schreibende Prozess sieht dabei Schnittstellen für die An- und Abmeldung von Beobachtern vor, während diese über eine

[13] Engl.: Tag

[14] Auch unter Publish/Subscribe bekannt.

Update-Schnittstelle Benachrichtigungen über die Zustandsänderung des beobachteten Objektes erhalten. Der Vorteil dieses Entwurfsmusters ist, dass die Komponenten untereinander lose gekoppelt sind und somit eine hohe Änderbarkeit dieser erreicht werden kann.

Abbildung 15 zeigt ein entsprechendes Sequenzdiagramm, in dem die Anmeldung und der Versand der Update-Nachricht dargestellt sind. Nachdem der Leser die Kommunikation initialisiert und dem Schreiber, mittels der Nachricht „attach“ und der übergebenen Shared Memory ID (kurz: shmID), mitgeteilt hat, von welchem Shared Memory er über Zustandsänderungen informiert werden möchte, erhält er entsprechende Update-Benachrichtigungen nach jedem Schreibvorgang durch den schreibenden Prozess. Die Update-Nachricht enthält als Informationen die Shared Memory ID, den Offset zum geschriebenen Slot sowie den gültigen Zeitstempel. Nach Erhalt der Update-Benachrichtigung kann der Leser die geschriebenen Daten des Slots lesen, solange der virtuelle Zeitstempel des Shared Memories kleiner als der gültige Zeitstempel des Slots ist. Die Abmeldung erfolgt analog durch Übergabe der Shared Memory ID mittels der Nachricht „detach“. Der Erfolg bzw. Misserfolg der An- und Abmeldung wird dem Leser durch eine entsprechende Antwortnachricht mitgeteilt.

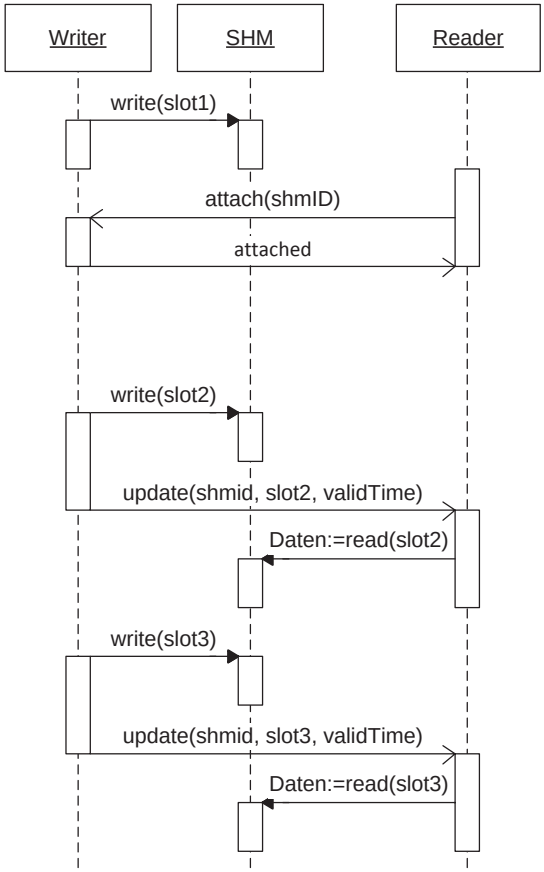

Abbildung 15 Anmeldung und Versand einer Update-Nachricht zur Synchronisation

3.2 Anforderungen an das Monitoring-System

Die einzuführende Systemüberwachung soll ein unterstützendes Werkzeug für Entwickler der Videoverarbeitungskette darstellen, mithilfe dessen Abläufe innerhalb der Prozesskette überwacht, visualisiert und analysiert werden.

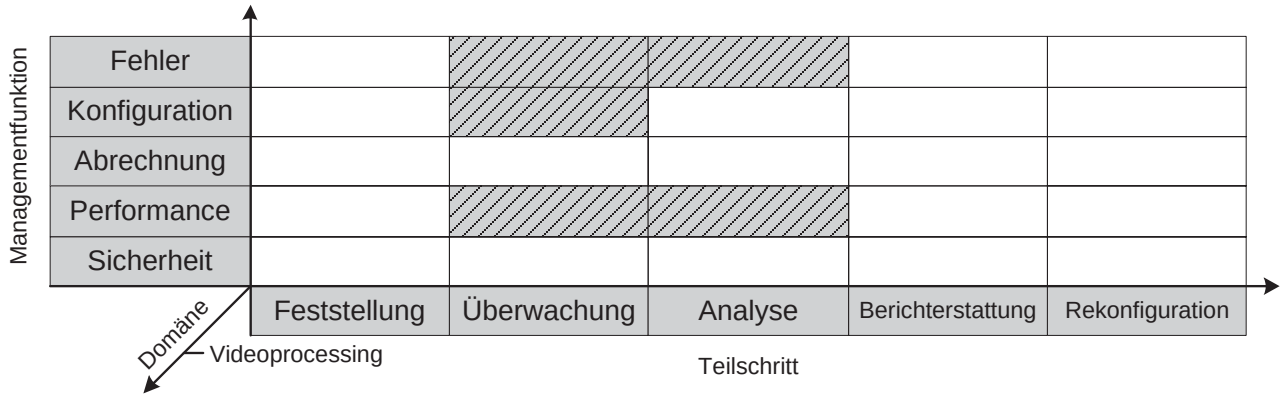

Abbildung 16 Angewandte Management-Matrix

Die grau gekennzeichneten Felder in Abbildung 16 sind die Teilschritte der Managementfunktionen, welche für die Verarbeitungskette zutreffen. Aufgrund des Einsatzbereiches ergeben sich spezielle Anforderungen aus den Bereichen des Performance-, Konfigurations- und Fehlermanagements. Nachfolgend werden diese Anforderungen detaillierter beschrieben.

3.2.1 Überwachungsanforderungen

Aus Abbildung 16 ist zu erkennen, dass sich die Überwachung des Systems zur Videoverarbeitung aus Funktionen des Fehler-, Konfigurations- und Performancemanagements zusammensetzt. Das System zur Videoverarbeitung kann je nach Anwendungsfall aus einem Zuspieler und einem Server zur Verarbeitung oder aus mehreren Zuspielgeräten und Server bestehen, um beispielsweise eine Redundanz des Systems zu schaffen. Daher müssen die Informationen, die zur Überwachung und der späteren Analyse benötigt werden, auch von unterschiedlichen Komponenten gesammelt werden. Diese Informationen werden zum Teil für die weitere Verarbeitung von mehreren Managementfunktionen benötigt. Aus diesem Grund werden die Überwachungsanforderungen im weiteren Verlauf nach den überwachten Komponenten unterschieden.

3.2.1.1 Anforderungen an die Serverüberwachung

Auf den überwachten Servern werden ausgewählte Informationen zu bestimmten Hardwareressourcen gesammelt. Dazu gehört der belegte Arbeitsspeicher durch die Anwendungen, die CPU- sowie die Netzwerk-Auslastung. Anhand der CPU-Auslastung lässt sich allgemein erkennen, ob die rechenintensiven Prozesse, wie der Video-Encoder, arbeiten oder ob es beispielsweise bis zu dieser Stufe zu einer Unterbrechung des Eingangsdatenstromes gekommen ist. Der Abbruch des Eingangsdatenstromes lässt sich, wie in Abbildung 17 zu sehen, auch anhand der Netzwerkauslastung erkennen, welche in diesem Fall um ein vielfaches abfällt. Die Informationen zur CPU- und Netzwerkauslastung sind sowohl für das Performance- als auch das Fehlermanagement notwendig.

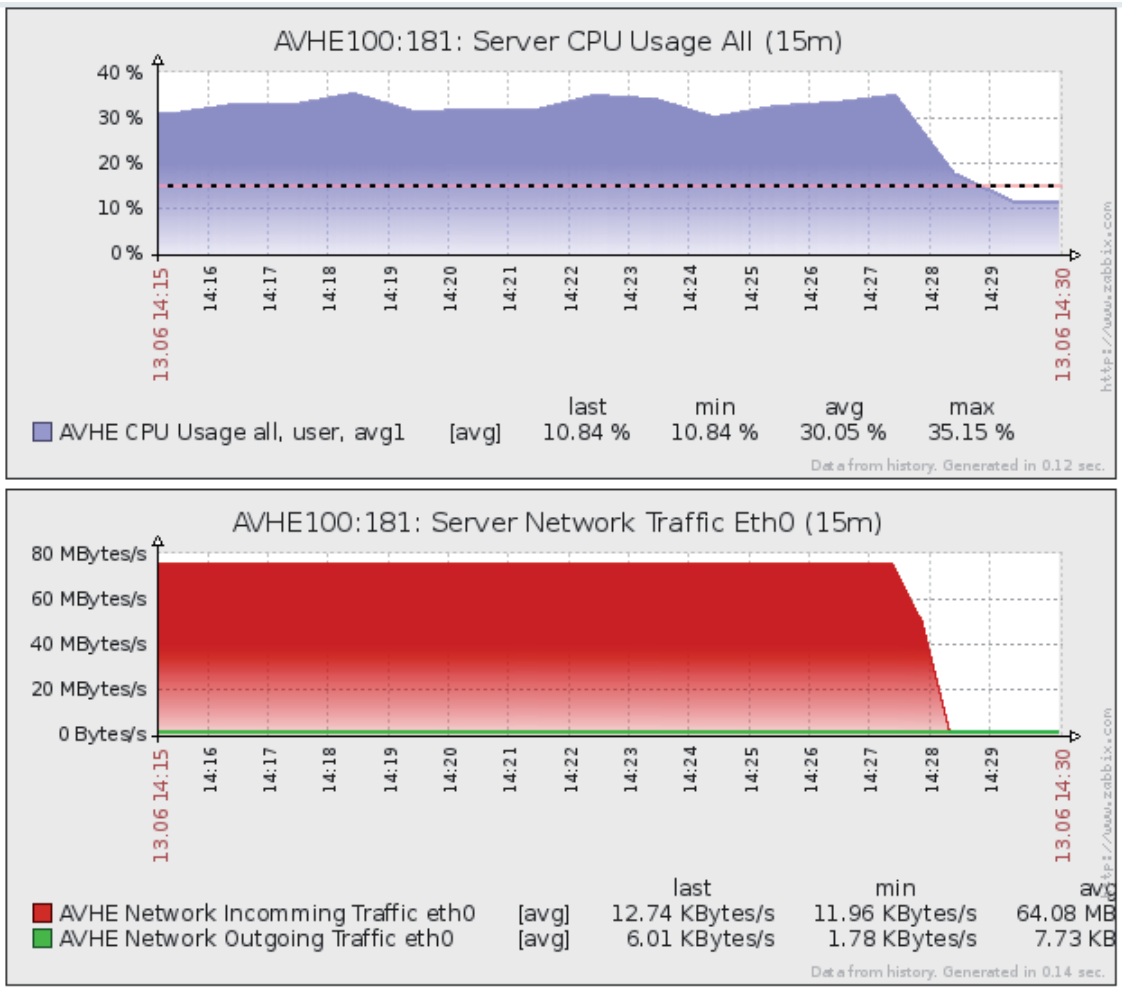

Abbildung 17 Ausfall des Eingangsdatenstromes

Andere Ressourcen wie z. B. die Festplattenkapazität sind weniger kritisch und müssen nicht primär überwacht werden. Das Gesamtsystem belegt bei einer bestimmten Konfiguration der Prozesskette einen definierten Betrag an Festplattenkapazität, d. h. es werden keine Prozessdaten auf dem Server dauerhaft gespeichert. Das System muss in seiner Hardwareausstattung entsprechend dimensioniert sein, um die vorgesehenen

Konfigurationen auch umsetzen zu können. Im Vergleich hierzu kann der verfügbare Speicher auf einem Datenbankserver aufgebraucht werden, da ständig Daten gespeichert werden müssen, um sie zu gegebener Zeit wieder abrufen zu können. Um rechtzeitig eingreifen zu können ist die Überwachung dieser Ressourcen in solch einem Fall durchaus sinnvoll und wichtig.

3.2.1.2 Anforderungen an die Anwendungsüberwachung

Im Fokus der Anwendungsüberwachung steht die Videoverarbeitungskette mit den genutzten Shared Memories. Diese dienen der Interprozesskommunikation zwischen den Verarbeitungsprozessen und enthalten, neben allgemeinen Informationen, sämtliche Daten, die von den Prozessstufen für die Verarbeitung benötigt werden. Die Shared Memories dienen der passiven Überwachung der Videoverarbeitungskette als Datenquelle und enthalten je nach Prozessstufe unterschiedliche Informationen (siehe Abschnitt 3.1.2.1). Mithilfe des bereits vorhandenen Konsolenprogrammes „Spucat“ können die Informationen eines Shared Memories ausgelesen und auf der Konsole als aufbereitete Zeichenkette ausgegeben werden. Um die Entwicklungszeit zu verkürzen sowie vorhandene Ressourcen und Know-how zu nutzen, wird dieses Programm als Konnektor eingesetzt. Bei einem aktiven Shared Memory erhält es periodische Update-Signale und bekommt auf diese Weise mitgeteilt, ab welchem Zeitpunkt neue Daten vorliegen. Daher erfolgt die Überwachung der Shared Memories auf passivem Wege (siehe Abschnitt 2.4.1.1) Um die Informationen weiter verarbeiten zu können, muss die Ausgabe der Zeichenketten zerlegt und in ganze Zahlen bzw. Gleitkommazahlen umgewandelt werden.

Neben den allgemeinen Meta-Daten der Shared Memories sind für das Konfigurationsmanagement die speziellen Meta-Daten interessant. Diese geben Auskunft über verschiedene Einstellungen. So sind bei einem Shared Memory, welches einen unkomprimierten Audiostrom enthält, die Abtastrate und die Anzahl der Kanäle gespeichert. Ein Shared Memory mit unkomprimiertem Video enthält unter anderem die Auflösung des Videos. Diese Informationen sollen durch den Entwickler einsehbar sein. Für die nachfolgende Analyse der Daten durch das Fehlermanagement sind die verschiedenen Zeitstempel des Transportstromes interessant (für Details zum Transportstrom siehe Abschnitt 2.6). So liegt beispielsweise der Grenzwert des zeitlichen Abstandes zweier aufeinanderfolgender PTS-Werte bei 700ms. Andernfalls liegt ein Presentation Timestamp Fehler (PTS_Error) vor. Ebenfalls von Bedeutung für das Fehlermanagement sind bestimmte Einträge in den Logdateien der Anwendungen. In den Dateien werden Meldungen mit unterschiedlichem Schweregrad abgelegt (siehe Abschnitt 2.5.3). Je nach Konfiguration werden diese Einträge durch das Fehlermanagement ausgewertet und dem Benutzer angezeigt oder ignoriert.

3.2.1.3 Anforderungen an die Netzwerküberwachung

Wird ein erweitertes System, bestehend aus mehreren Server und Zuspielern, zur Verarbeitung von einem oder mehreren Videoströmen verwendet, so ist der Einsatz von zusätzlicher Netzwerkhardware, wie z. B. Switch, notwendig. Um die Überwachung der Netzwerkkomponenten über eine gemeinsame Schnittstelle durchführen zu können, wird das Simple Network Management Protocol eingesetzt (siehe Abschnitt 2.5.1). Vorzugsweise soll dabei die aktuellste Version 3 des Protokolls unterstützt werden, da dieses im Gegensatz zu vorherigen Versionen wesentliche Sicherheitsaspekte unterstützt. Des Weiteren wird mithilfe des ICMP Nachrichtentyps „Echo“ die Verfügbarkeit von Netzwerkkomponenten in regelmäßigen Abständen überprüft. Zu den überprüften Komponenten gehören die verwendeten Netzwerkkomponenten und Server, die nicht per SNMP oder einem proprietären Protokoll angesprochen werden können.

3.2.2 Analyseanforderungen

Die Analyse und anschließende Interpretation der gesammelten Informationen obliegt dem Entwickler selbst, da nur er die gewonnen Informationen bewerten kann. Er muss gegebenenfalls das weitere Vorgehen bestimmen, um die Ursache eines Fehlers genau einzugrenzen. Bei dieser Aufgabe wird er durch das Überwachungssystem unterstützt. Dabei reicht es nicht aus, die gewonnen Daten in Form einer Tabelle wiederzugeben. Das Monitoring-System muss in der Lage sein, die Daten ansprechend visuell darstellen zu können.

Die gewonnenen Messwerte werden innerhalb eines Linien- bzw. Flächendiagrammes über den zeitlichen Verlauf dargestellt. Da sich einzelne Datenfehler im Transportstrom nicht auf den Verlauf der nachfolgenden Daten auswirken, ist es wichtig, dass die eingegangenen Daten in einem geeigneten Intervall roh dargestellt werden, d. h. die Graphen stellen die Messwerte ohne weitere Verarbeitung, z. B. Mittelwertbildung, dar. Nur so können einzelne Fehler bzw. Ausreißer im Kurvenverlauf erkannt werden, da sich einzelne abweichende Messwerte nur unmerklich auf den Verlauf der gesamten Kurve auswirken. Das geeignete Intervall ergibt sich in diesem Zusammenhang aus der Summe der darstellbaren Messpunkte x durch die Anzahl vps der pro Sekunde gewonnenen Messwerte.

$$t_I = \frac{\sum x}{vps} \qquad \text{(Formel 2)}$$

Das folgende Beispiel soll diesen Zusammenhang verdeutlichen. Wird ein Diagramm als Rastergrafik dargestellt, so besteht es aus einer definierten Pixelanzahl für Höhe und Breite. Für die Darstellung der Messwerte auf der Abszissenachse steht abzüglich Beschriftung und Legende eine Nettobreite in Form von Pixeln zur Verfügung. Beträgt

diese Nettobreite 1800 Pixel und werden pro Sekunde 60 Messwerte erfasst, die jeweils ein Pixel benötigen, so ergibt sich ein maximales Intervall von 30 Sekunden:

$$t_I = \frac{1800\text{Px}\cdot\text{s}}{60\text{Px}} = 30\text{s} \quad \text{(Formel 3)}$$

Wenn die Messwerte einen sprunghaften Verlauf annehmen, d. h. wenn benachbarte Messwerte sich um ein Vielfaches unterscheiden, kann es zu einer Stauchung bzw. Blockbildung des Kurvenverlaufes, wie in Abbildung 18 zu sehen, kommen.

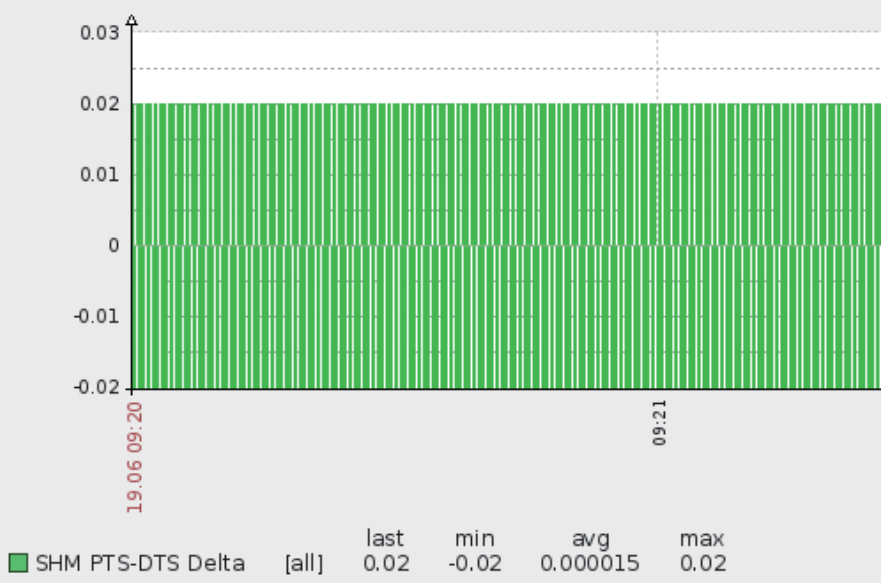

Abbildung 18 Stauchung des Graphen

In diesem Fall ist ein kleineres Intervall zu wählen, um den Kurvenverlauf wieder zu entzerren.

Um die Analyse der eingehenden Daten zu vereinfachen, werden Trigger im Fehlermanagement verwendet, die bei über- bzw. unterschreiten von definierten Grenzwerten auf dieses Ereignis reagieren. Im einfachsten Fall wird dem Entwickler durch eine Meldung angezeigt, dass ein Grenzwert erreicht wurde. Optional werden die definierten Grenzwerte innerhalb der Diagramme gekennzeichnet.

Alle eingegangen Informationen werden in Übersichten dargestellt, um einen Gesamtüberblick über diese erhalten zu können. Hierzu werden ausgewählte Graphen in einer verkleinerten Darstellung mit Daten aus dem gleichen Zeitraum in Tabellenform dargestellt. Neben der graphischen Darstellung der eingegangenen Daten sind diese in Tabellenform abrufbar. Dies ist besonders für die Konfigurationsparameter wichtig, da diese über einen längeren Zeitraum nicht verändert werden.

3.2.3 Weitere Anforderungen

Um den Aufwand für Softwareverwaltung und -pflege so gering wie möglich zu halten, sind die gesammelten Daten über eine Weboberfläche, ein sogenanntes Web-Frontend, zugänglich. Dies erspart die Bereitstellung und Wartung einer entsprechenden Desktopanwendung. Vorzugsweise erfolgt die Konfiguration des Monitoring-Systems ebenfalls zu weiten Teilen über diese Weboberfläche. Hierzu zählt die Konfiguration der dynamischen Parameter, wie z. B. der Name des zu überwachenden Shared Memories. Optional werden die Agenten sowie der Managementserver über diese Oberfläche konfiguriert. Dies verringert das notwendige Hintergrundwissen zur Installation und Wartung des Überwachungssystems wesentlich. Da ein Serverbetriebssystem in der Regel über keine Oberfläche verfügt, erfolgt andernfalls die Konfiguration lokal auf dem Server über einen Zugang zum Betriebssystem z. B. über Secure Shell (SSH). Dies erfordert entsprechendes Hintergrundwissen im Umgang mit dem Betriebssystem sowie über das eigentlich zu konfigurierende Monitoring-System.

Die Ausführung der Videoverarbeitungskette hat oberste Priorität und darf durch die Überwachungsmaßnahmen nicht beeinträchtigt werden. Diesbezüglich dürfen die Überwachungsanwendungen den IT-Server nur minimal belasten. Der Einsatz von Interpreter-Sprachen zur Erstellung von Erweiterungen und zusätzlichen Programmen ist daher auf ein Minimum zu reduzieren. Des Weiteren erfolgt lediglich das Sammeln der notwendigen Daten auf dem überwachten Server selbst. Die weitere Verarbeitung sowie die Speicherung der Daten erfolgt auf einem separaten Server, dem Managementserver. Die Monitoring-Anwendung folgt dem Agenten-Manager-Modell aus Abbildung 19.

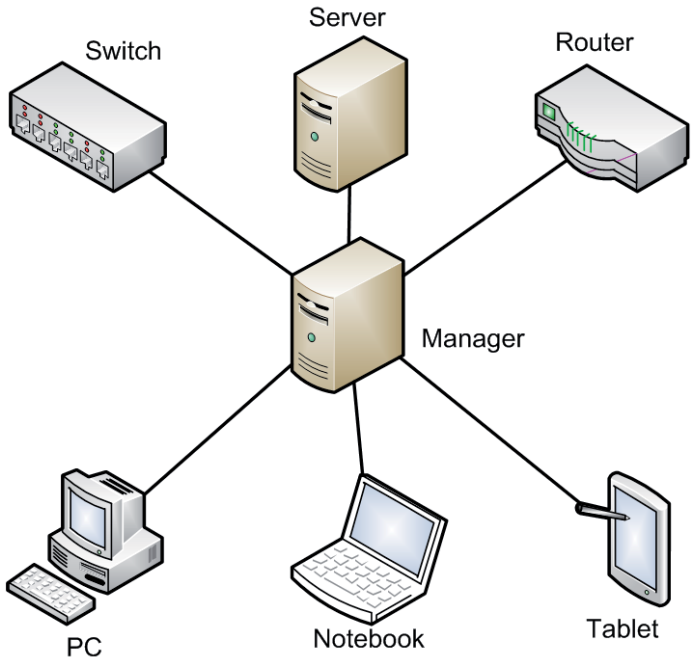

Abbildung 19 Agenten-Manager-Modell

In diesem Modell ist ein Manager für das Sammeln und Verarbeiten der Informationen zuständig. Die Agenten auf den zu überwachenden Geräten stellen dabei den Zugang zur Informationsquelle her.

Um einen möglichst hohen Investitionsschutz zu erreichen, ist eine Monitoring-Software einzusetzen, die einen hohen Entwicklungsstand aufweist und aktiv gepflegt sowie weiterentwickelt wird. Auf diese Weise kann sichergestellt werden, dass die Software auch in Zukunft weiter Anwendung findet und nicht mangels Unterstützung durch die Entwickler eingestellt wird. Hierzu ist auf eine gepflegte Dokumentation, Frequently Asked Questions (FAQ) und eine kommerzielle Unterstützung zu achten. Der Einsatz einer populären Open-Source-Software bringt eine aktive Gemeinschaft von Entwicklern und Nutzern mit sich. Daher ist eine Monitoring-Software zu wählen, die unter einer freien Lizenz wie z. B. der GPL v2[15] veröffentlicht ist. Ein weiterer Vorteil des Einsatzes von Open-Source Software ist, dass die Quellen eingesehen werden können. So kann die Software nach eigenen Bedürfnissen angepasst oder erweitert werden.

Für eine zukünftige vertiefende Nutzung des Monitoring ist eine Schnittstelle bzw. API vorzusehen. Im einfachsten Fall werden über diese Schnittstelle die gespeicherten Informationen abgerufen, sodass diese durch externe Komponenten verarbeitet werden können. Dies kann beispielsweise über die Verwendung von einem geläufigen Datenbankmanagementsystem (DBMS), wie z. B. MySQL, zur Speicherung der Daten erfolgen. Dabei ist darauf zu achten, dass die Struktur der Datenbank ausreichend dokumentiert ist. Ein anderer Fall sieht vor, dass das Monitoring-System über eine Schnittstelle gesteuert werden kann, d. h. die Konfiguration kann über diese vorgenommen werden. Über eine solche Schnittstelle kann eine individuelle Oberfläche implementiert werden, die auf die speziellen Bedürfnisse des Arbeitsumfeldes zugeschnitten ist. Eine genaue Festlegung auf eine bestimmte Schnittstelle ist zum derzeitigen Zeitpunkt nicht möglich.

[15] GNU General Public License Version 2, siehe: http://www.gnu.org/licenses/gpl.html

3.3 Bewertung der verfügbaren Systeme

3.3.1 Bewertungsmaßstab

Die beschriebenen Anforderungen aus dem vorhergehenden Abschnitt 3.2 flossen in die nachfolgende Tabelle 4 mit den Prüfkriterien:

Tabelle 4 Prüfkriterien

Anforderungs-Gruppe	**Prüfkriterium**	**Punkteverteilung (Gesamt: 45 Pkt.)**
1) Überwachung	Server:	6
	– CPU-Auslastung	2
	– Netzwerk-Auslastung	2
	– Speicherbelegung	2
	Anwendung:	8
	– Passive Überwachung	3
	– Verarbeitung Zeichenketten	3
	– Auswertung Logdateien	2
	Netzwerk:	3
	– ICMP Echo-Nachricht	1
	– SNMP	2
2) Analyse	Darstellungsform:	8
	– Tabellen	3
	– Graphen	3
	– Übersichten	2
	Verarbeitung:	5
	– Sekündliche Skaleneinteilung	3
	– Grenzwert-Trigger	2
3) Weiteres	Technische Merkmale:	7
	– Konfiguration über Web-Frontend	2
	– Native Hostanwendungen	2
	– Agenten-Manager Architektur	2
	– Externe Schnittstellen	1
	Nicht-technische Merkmale:	8
	– Handhabung Webfrontend/Graphen	2
	– Dokumentation/FAQ/Anleitungen	2
	– Aktive Gemeinschaft/Support	2
	– Aufwand Installation/Wartung	2

Um eine Auswahl an zu bewertenden Überwachungssystemen aus den am Markt verfügbaren Systemen zu erhalten, sind diese anhand folgender Kriterien zu selektieren:

- Veröffentlichung unter einer freien Lizenz wie z. B. der GPL v2
- Verarbeitung von Gleitkommazahlen und ganzen Zahlen
- Datendarstellung innerhalb von Linien- bzw. Flächendiagramm
- Zugang zu eingegangenen Daten über Web-Frontend

Die auf diese Weise selektierten Systeme werden anschließend anhand des erstellten Bewertungsmaßstabes miteinander verglichen und bewertet. Die Bewertung findet dabei auf Unterstützung der aufgeführten Prüfkriterien statt. Diese Prüfkriterien sind analog den beschriebenen Anforderungen aus Abschnitt 3.2 gruppiert und bestehen aus logischen Untergruppen mit zusammengehörigen Kriterien. Jedem Prüfkriterium ist eine bestimmte Anzahl an zu erreichenden Punkten zugeordnet. Die Anzahl der Punkte der Untergruppen ergibt sich aus der Summe der Punkte der untergeordneten Kriterien. Beispielsweise setzt sich die Punkteanzahl der Serverkriterien aus den Punkten der Speicherbelegung sowie der CPU- und Netzwerkauslastung zusammen.

Die maximal zu erreichenden Punkte eines Prüfkriteriums reichen von einem Punkt bis drei Punkten und entsprechen somit einer bestimmten Priorität.

Drei Punkte entsprechen dabei der höchsten Priorität. Prüfkriterien, welche Eigenschaften der Überwachungsanwendung beschreiben, die für die Überwachung der Videoverarbeitungskette unbedingt erforderlich sind, werden mit maximal drei Punkten bewertet. Dabei sind folgende Punkteverteilungen aus Tabelle 5 möglich.

Tabelle 5 Punkteverteilung bei höchster Priorität

Punkte	Beschreibung
0	Eigenschaft wird durch die Überwachungsanwendung nicht unterstützt
1	Eigenschaft wird durch die Überwachungsanwendung nur rudimentär unterstützt
2	Eigenschaft wird durch die Überwachungsanwendung unterstützt, muss jedoch für die geplante Verwendung im geringen Maße angepasst werden
3	Eigenschaft wird durch die Überwachungsanwendung voll unterstützt

Ist eine Überwachungsanwendung in der Lage, Zeichenketten zu empfangen und abzuspeichern, kann aus dieser jedoch keine weiteren Daten entnehmen, wird dieses beispielsweise mit einem Punkt für das Kriterium „Verarbeitung Zeichenketten“ bewertet, da die Extraktion der Daten notwendig ist, aber nicht durch geringe Maßnahmen ergänzt werden kann.

Zwei Punkte entsprechen einer mittleren Priorität für Eigenschaften der Überwachungsanwendung, die nicht unbedingt erforderlich sind, jedoch die Überwachung der Videoverarbeitungskette sinnvoll ergänzen. Dabei sind folgende Punkteverteilungen aus Tabelle 5 möglich.

Tabelle 6 Punkteverteilung bei mittlerer Priorität

Punkte	Beschreibung
0	Eigenschaft wird durch die Überwachungsanwendung nicht unterstützt
1	Eigenschaft wird durch die Überwachungsanwendung mit Einschränkungen unterstützt
2	Eigenschaft wird durch die Überwachungsanwendung voll unterstützt

Unterstützt beispielsweise eine Überwachungsanwendung SNMP nicht in der aktuellen Version 3, ist dieses für das Prüfkriterium SNMP mit einem Punkt zu bewerten, wenn es SNMP in einer vorhergehenden Version unterstützt.

Mit zwei Punkten werden ebenfalls nicht-technische Merkmale aus der dritten Gruppe bewertet. Hierbei ist zu beachten, dass die Punkteverteilung aus Tabelle 6 keine Anwendung findet, sondern diese nach ihrer positiven Ausprägung bewertet werden, siehe Tabelle 7. D. h. es wird auf eine intuitive Handhabung der Weboberfläche bzw. der Graphen, eine umfangreiche Dokumentation mit Beispielen, eine aktive Gemeinschaft von Nutzern und Entwicklern sowie einen geringen Aufwand für Installation und Wartung geachtet.

Tabelle 7 Punkteverteilung bei nicht-technischen Merkmalen

Punkte	Beschreibung
0	Merkmal ist kaum ausgeprägt oder weist negative Tendenzen auf
1	Merkmal ist im Allgemeinen befriedigend ausgeprägt
2	Merkmal ist im höchsten Maße ausgeprägt

Ein Merkmal mit negativen Tendenzen kann beispielsweise „aktive Gemeinschaft/Support“ sein, wenn keine Aktivitäten durch Nutzer und Entwickler mehr zu verzeichnen sind.

Ein Punkt entspricht einer niedrigen Priorität für Eigenschaften der Überwachungsanwendung, die für eine zukünftige erweiterte Verwendung dieser interessant sind. Diese Kriterien werden mit einem Punkt bewertet, wenn die Eigenschaften in einer bestimmten Form unterstützt werden.

Zur abschließenden Bewertung der Überwachungssysteme werden folgende Noten aus Tabelle 8 vergeben:

Tabelle 8 Bewertungsnoten

Note	Bezeichnung	Erläuterung
1	sehr gut	hervorragende Leistungen
2	gut	Leistungen, die erheblich über den durchschnittlichen Anforderungen liegen
3	befriedigend	Leistungen, die durchschnittlichen Anforderungen entsprechen
4	ausreichend	Leistungen, die trotz ihrer Mängel noch den Anforderungen genügen
5	nicht ausreichend	Leistungen, die wegen erheblicher Mängel den Anforderungen nicht mehr genügen

Mit der Note 4 (ausreichend) und besser werden Überwachungssysteme bewertet, die mindestens 15 Punkte erreichen. Dies wird durch Erfüllung aller Kriterien mit der höchsten Priorität erreicht. Werden weniger als 15 Punkte erreicht, so wird das Überwachungssystem mit der Note 5 (nicht ausreichend) bewertet.

Die erreichte Gesamtpunktzahl wird durch einen linearen Punkte-Noten-Schlüssel in eine entsprechende Note zwischen 1 und 4 umgewandelt. Hierzu wird die nachfolgende Gleichung verwendet:

$$\mathrm{N} = \mathrm{N_{Min}} - (\mathrm{N_{Min}} - \mathrm{N_{Max}}) \cdot \frac{\mathrm{P} - \mathrm{P_{Min}}}{\mathrm{P_{Max}} - \mathrm{P_{Min}}} \qquad \text{(Formel 4)}$$

Aus dieser Formel geht die Note N aus der erreichten Punktzahl P hervor. Um eine Note zwischen der Mindestnote $\mathrm{N_{Min}}$ und der Maximalnote $\mathrm{N_{Max}}$ zu erhalten, muss die erreichte Punktzahl zwischen den Mindestpunkten $\mathrm{P_{Min}}$ und den Maximalpunkten $\mathrm{P_{Max}}$ liegen.

Für die Berechnung der Noten mithilfe der oben genannten Gleichung sind folgende Werte festgelegt:

$$\mathrm{P_{Min}} = 15\ \mathrm{Pkt.} \qquad \mathrm{P_{Max}} = 45\ \mathrm{Pkt.}$$
$$\mathrm{N_{Min}} = 4 \qquad \mathrm{N_{Max}} = 1$$

Aus diesen Werten ergibt sich die folgende aufgelöste Gleichung:

$$\mathrm{N} = 4 - \frac{\mathrm{P} - 15}{10} \qquad \text{(Formel 5)}$$

Zur differenzierten Bewertung werden ab der Note 4 Zehntelnoten vergeben (4,0; 3,9; 3,8 usw.) und die berechneten Noten auf eine Dezimalstelle gerundet.

3.3.2 Ganglia Monitoring-System

Tabelle 9 Bewertung Ganglia Monitoring-System

Ganglia Core 3.4.0 & Web 3.5.2	Punkte	Bemerkung
1) Überwachung		
Server		
CPU-Auslastung	2/2	
Netzwerk-Auslastung	2/2	
Speicherbelegung	2/2	
Anwendung		
Passive Überwachung	3/3	Mit Gmond als Agent und Gmetric
Verarbeitung Zeichenketten	1/3	Plug-in Erstellung notwendig
Auswertung Logdateien	2/2	Durch Module verfügbar
Netzwerk		
ICMP Echo-Nachricht	1/1	Durch Integration eines Plug-ins
SNMP	0/2	Keine SNMP Unterstützung
2) Analyse		
Darstellungsform		
Tabellen	2/3	Nur über Datenexport
Graphen	3/3	
Übersichten	2/2	
Verarbeitung		
Sekündliche Skaleneinteilung	2/3	Anpassung notwendig
Grenzwert-Trigger	2/2	Durch Integration eines Plug-ins
3) Weiteres		
Technische Merkmale		
Konfiguration über Web-Frontend	0/2	Ausschließlich über Dateien
Native Hostanwendungen	2/2	
Agenten-Manager Architektur	2/2	
Externe Schnittstellen	1/1	Abfrage über Datenmanagement
Nicht-technische Merkmale		
Handhabung Webfrontend/Graphen	2/2	Zoomen in Graphen möglich
Dokumentation/FAQ/Anleitungen	1/2	Dokumentation unvollständig
Aktive Gemeinschaft/Support	1/2	Support nur über Maillisten
Aufwand Installation/Wartung	1/2	Aktuelle Installation nur von Quellen
Summe	34/45	

Das Ganglia Monitoring-System wird von verschiedenen namenhaften Firmen zur Überwachung von deren Servern und Clustern verwendet. Es besteht aus zwei Paketen, zum einem dem Monitor Core Paket[16], welches die notwendigen Komponenten zur Überwachung beinhaltet, sowie dem Web Paket[17], welches die grafische Weboberfläche bereitstellt. Das Monitor Core Paket beinhaltet den Agenten, den sogenannten Ganglia Monitor Daemon (kurz: Gmond), welcher verschiedene Module zur Überwachung von Serverressourcen und Anwendungen einsetzt, sowie den Manager namens Ganglia Metadata Daemon (kurz: Gmetad). Der Agent kann mithilfe von eigenen Modulen erweitert werden. Verfügbar sind verschiedene Erweiterungen zur Überwachung von Logdateien sowie zum Versenden von ICMP Nachrichten. Das Überwachen von Netzwerkgeräten mittels SNMP wird von Ganglia nicht nativ unterstützt und muss über externe Anwendungen erfolgen. Die gewonnen Daten dieser Anwendungen können dennoch von Ganglia gespeichert und visualisiert werden. Zum Senden der Daten an den Manager steht ein Kommandozeilenprogramm namens Ganglia Metric Tool (kurz: Gmetric) zur Verfügung. Ganglia unterstützt hierfür verschiedene Datentypen für Gleitkommazahlen, ganze Zahlen und Zeichenketten. Die weitere Verarbeitung der Zeichenketten ist jedoch nicht möglich.

Die gewonnen Daten werden vom Manager in einer Round-Robin Datenbank (RRD) abgelegt. Die Daten können vom Web-Frontend innerhalb verschiedener Graphen visualisiert werden. Durch Markierung eines bestimmten Ausschnittes des Graphen kann in diesen Bereich hinein gezoomt werden, um Details erkennen zu können. Eine tabellarische Ansicht der eingegangenen Daten fehlt, diese können jedoch als CSV-Datei[18] exportiert werden. Zur Darstellung der eingegangenen Daten in einem sekündlichen Zeitbereich ist die Anpassung der Datenbank sowie der Weboberfläche notwendig.

Zur Installation der aktuellen Version, muss diese zuvor aus den Quelldateien erstellt worden sein. Die Konfiguration des Managers, Agenten und Web-Frontends erfolgt ausschließlich über die Konfigurationsdateien der Komponenten. Damit verbunden ist ein höherer Aufwand zur Veränderung der Dateien bei großen Clustern. Unterstützend bei der Installation und Konfiguration existieren verschiedene Dokumente und Artikel. Die Ganglia Dokumentation selbst ist inhaltlich unvollständig und in Teilen unstrukturiert. Unterstützung durch die Entwickler wird ausschließlich über Mailing-Listen geboten.

---

[16] Release 3.4.0 vom 24.05.2012

[17] Release 3.5.2 vom 16.07.2012

[18] Comma-Separated Values (CSV)

3.3.3 Munin

Tabelle 10 Bewertung Munin Monitoring-System

Munin 2.0.2	Punkte	Bemerkung
1) Überwachung		
Server		
CPU-Auslastung	2/2	
Netzwerk-Auslastung	2/2	
Speicherbelegung	2/2	
Anwendung		
Passive Überwachung	0/3	Nur aktive Abfragen durch Master
Verarbeitung Zeichenketten	1/3	Plug-in Erstellung notwendig
Auswertung Logdateien	1/2	Über Plug-in verfügbar
Netzwerk		
ICMP Echo-Nachricht	1/1	Über Plug-in verfügbar
SNMP	2/2	Über Plug-in verfügbar
2) Analyse		
Darstellungsform		
Tabellen	1/3	Erstellung einer Ansicht notwendig
Graphen	3/3	
Übersichten	2/2	
Verarbeitung		
Sekündliche Skaleneinteilung	2/3	Anpassung notwendig
Grenzwert-Trigger	1/2	Plug-in muss Trigger unterstützen
3) Weiteres		
Technische Merkmale		
Konfiguration über Web-Frontend	0/2	Ausschließlich über Dateien
Native Hostanwendungen	2/2	In Perl implementiert
Agenten-Manager Architektur	2/2	
Externe Schnittstellen	1/1	Abfrage über Datenmanagement
Nicht-technische Merkmale		
Handhabung Webfrontend/Graphen	2/2	Zoomen in Graphen möglich
Dokumentation/FAQ/Anleitungen	2/2	
Aktive Gemeinschaft/Support	1/2	Support nur über Maillisten
Aufwand Installation/Wartung	1/2	Aktuelle Installation nur von Quellen
Summe	31/45	

Munin zeichnet sich besonders durch eine hohe Flexibilität aus, die durch ein ausgeprägtes Plug-in-System erreicht wird. In der aktuellen Version[19] unterstützt Munin durch die mitgelieferten Metriken das Überwachen von verschiedenen Ressourcen eines Servers sowie der Anwendungen. Der Agent, die sogenannte Munin Node, unterstützt dabei alle gängigen Linux Distributionen, BSD sowie proprietäre Betriebssysteme wie z. B. Windows oder OS X. Eine Vielzahl weiterer Metriken sind in Form von Plug-ins über das Projekt Archiv[20] verfügbar. So ist es möglich, gängige Datenbankmanagementsysteme oder Logdateien von Anwendungen zu überwachen. Netzwerkgeräte können per mitgeliefertem SNMP Plug-in überwacht werden. Dieses muss zuvor auf die gewünschte Anwendung angepasst werden. Munin bietet hierzu ein Skript, welches bei der Konfiguration des Plug-ins unterstützt. Zur Auswertung von Zeichenketten steht kein Plug-in zur Verfügung. Hierzu ist die Implementierung eines eigenen Plug-ins notwendig. Die Erweiterung bzw. Implementierung von Plug-ins kann in der Skriptsprache Perl, in der Munin implementiert ist, erfolgen. Um rechenintensive Algorithmen effizient umzusetzen, können diese z. B. in der Programmiersprache C implementiert werden und durch Perl als Modul ausgeführt werden.

Der Munin Manager, in der Dokumentation auch Master genannt, fragt in einem festgelegten Intervall die Daten bei den Agenten ab und speichert eingegangene Daten in der angebunden Round-Robin Datenbank ab. Die passive Überwachung des Servers bzw. der Anwendungen ist nicht möglich. Die eingegangenen Daten werden innerhalb verschiedener Graphen visualisiert, eine tabellarische Ansicht fehlt jedoch. Das Hineinzoomen in einen bestimmten Ausschnitt des Graphen wird von Munin ebenfalls unterstützt. Da Munin auf das gleiche Datenbankmanagementsystem wie Ganglia zurückgreift, ist die Anpassung der Datenbankkonfiguration zur Darstellung der Daten im Sekundenbereich notwendig. Die Messwerte können auf Einhaltung definierter Grenzwerte überwacht werden, soweit dies von dem messenden Plug-in unterstützt wird.

Zur Installation von Munin stehen unter verschiedenen Linux Distributionen Installationspakete zur Verfügung, die jedoch nicht die aktuellste Version beinhalten. Das Einrichten des Masters und der Nodes erfolgt ausschließlich über Konfigurationsdateien, die sich auf den überwachten Hosts bzw. dem Master befinden. Die Projekt-Website stellt hierzu unterstützend Hilfen, ein FAQ und eine umfangreiche Dokumentation bereit. Unterstützung durch die Entwickler kann lediglich über Mailinglisten bezogen werden.

---

[19] Release 2.0.2 vom 29.06.2012

[20] Siehe: https://github.com/munin-monitoring/contrib/

3.3.4 Cacti

Tabelle 11 Bewertung Cacti Monitoring-System

Cacti 0.8.8a	**Punkte**	**Bemerkung**
1) Überwachung		
Server		
CPU-Auslastung	2/2	Als Skript verfügbar
Netzwerk-Auslastung	2/2	Als Skript verfügbar
Speicherbelegung	2/2	Als Skript verfügbar
Anwendung		
Passive Überwachung	0/3	Nur aktive Abfragen durch Poller
Verarbeitung Zeichenketten	1/3	Plug-in Erstellung notwendig
Auswertung Logdateien	2/2	Als Plug-in verfügbar
Netzwerk		
ICMP Echo-Nachricht	1/1	
SNMP	2/2	
2) Analyse		
Darstellungsform		
Tabellen	1/3	Erstellung einer Ansicht notwendig
Graphen	3/3	
Übersichten	2/2	
Verarbeitung		
Sekündliche Skaleneinteilung	3/3	Als Plug-in verfügbar
Grenzwert-Trigger	2/2	
3) Weiteres		
Technische Merkmale		
Konfiguration über Web-Frontend	2/2	
Native Hostanwendungen	1/2	Nur über externe Anwendungen
Agenten-Manager Architektur	2/2	
Externe Schnittstellen	1/1	Abfrage über Datenmanagement
Nicht-technische Merkmale		
Handhabung Webfrontend/Graphen	2/2	Zoomen in Graphen möglich
Dokumentation/FAQ/Anleitungen	2/2	
Aktive Gemeinschaft/Support	2/2	Support über Foren und Maillisten
Aufwand Installation/Wartung	1/2	Abhängig von anderen Programmen
Summe	36/45	

Cacti ist eine Monitoring-Software, die besonders auf die Überwachung von Netzwerkkomponenten ausgelegt ist. In der aktuellen Version[21] wird daher auf einen eigenen Agenten verzichtet. Stattdessen werden die Daten von SNMP-Agenten der Netzwerkgeräte bezogen. Server und Anwendungen können über die Ausführung von Skripten überwacht werden oder mithilfe eines zusätzlich installierten SNMP-Agenten, wie z. B. Net-SNMP[22]. Der sogenannte Poller fragt dabei in einem festgelegten Zeitraum die Daten aktiv bei den überwachten Geräten ab. Eine passive Überwachung der Geräte ist mithilfe von Cacti nicht möglich. Für die Überwachung der Speicherbelegung sowie der CPU- und Netzwerkauslastung werden entsprechende Skripte mitgeliefert. Darüber hinaus lässt sich Cacti mithilfe von weiteren verfügbaren Skripten und Plug-ins erweitern, sodass auch die Überwachung von Log-Dateien ermöglicht wird. Eine Verarbeitung von Zeichenketten ist nur mittels Implementierung von passenden Skripten möglich.

Eingehende numerische Daten werden innerhalb einer Round-Robin Datenbank abgespeichert. Dabei kommt die gleiche Anwendung wie bei Ganglia oder Munin zum Einsatz. Mithilfe dieser ist auch die Einrichtung von Triggern möglich, um die Einhaltung von definierten Grenzwerten sicherzustellen. Die gespeicherten Daten können innerhalb benutzerdefinierten Graphen visualisiert werden. Dabei stehen verschiedene Diagrammtypen zur Verfügung, deren Parameter über die Weboberfläche konfiguriert werden. Eine Zoom-Funktion erlaubt es, einen markierten Bereich des gewählten Graphen hervorzuheben. Die Darstellung der eingegangen Daten innerhalb einer Tabelle steht jedoch nicht zu Verfügung und muss gegebenenfalls ergänzt werden. Mithilfe eines Real-Time Plug-ins ist die Darstellung der Daten innerhalb eines kleinen Intervalls möglich.

Cacti benötigt für den Betrieb eine Reihe von zusätzlichen Anwendungen. Neben dem RRD Tool zur Datenspeicherung und einem Webserver für die Visualisierung wird ein MySQL Server benötigt, um die angelegten Konfigurationen speichern zu können. Hilfe bei der Installation der Monitoring-Software findet sich dabei innerhalb der umfangreichen Dokumentation. Diese umfasst auch Anleitungen zur Einrichtung eines Net-SNMP Agenten auf einem zu überwachenden Server. Weitere Unterstützung kann über das aktive Forum bezogen werden.

[21] Release 0.8.8a vom 29.04.2012

[22] Siehe: http://www.net-snmp.org/

3.3.5 Zabbix

Tabelle 12 Bewertung Zabbix Monitoring-System

Zabbix 2.0.1	Punkte	Bemerkung
1) Überwachung		
Server		
CPU-Auslastung	2/2	
Netzwerk-Auslastung	2/2	
Speicherbelegung	2/2	
Anwendung		
Passive Überwachung	3/3	Durch Zabbix Agenten realisierbar
Verarbeitung Zeichenketten	1/3	Erweiterung notwendig
Auswertung Logdateien	2/2	Durch Zabbix Agenten möglich
Netzwerk		
ICMP Echo-Nachricht	1/1	
SNMP	2/2	Unterstützt SNMPv3
2) Analyse		
Darstellungsform		
Tabellen	3/3	
Graphen	3/3	
Übersichten	2/2	
Verarbeitung		
Sekündliche Skaleneinteilung	2/3	Anpassung notwendig
Grenzwert-Trigger	2/2	
3) Weiteres		
Technische Merkmale		
Konfiguration über Web-Frontend	2/2	Nur Grundkonfiguration in Dateien
Native Hostanwendungen	2/2	Agent und Server in C geschrieben
Agenten-Manager Architektur	2/2	
Externe Schnittstellen	1/1	API als JSON-RPC implementiert
Nicht-technische Merkmale		
Handhabung Webfrontend/Graphen	2/2	Unterstützt zoomen in Graphen
Dokumentation/FAQ/Anleitungen	2/2	Umfangreiche Dokumentation
Aktive Gemeinschaft/Support	2/2	Bietet kommerziellen Support
Aufwand Installation/Wartung	1/2	Aktuelle Installation nur von Quellen
Summe	41/45	

Zabbix bietet in der aktuellen Version[23] eine Vielzahl unterstützter Informationsquellen, Metriken und Darstellungsformen. Der größte Teil der unterstützten Metriken wird dabei durch den Agenten bereitgestellt, der alle gängigen Betriebssysteme, wie z. B. verschiedene Linux Distributionen, Windows oder BSD unterstützt. Der Agent wird auf dem zu überwachenden Server (auch Host genannt) installiert und ist in der Lage, dessen Hardware als auch die darauf laufenden Anwendungen zu überwachen. Dabei kann die Datenerfassung sowohl passiv als auch aktiv erfolgen (siehe Abschnitt 2.4). Des Weiteren wird die Überwachung von Log- und Textdateien unterstützt. Um Daten aus den Zeichenketten zu extrahieren, muss auf externe Skripte oder Programme zurückgegriffen werden. Die Speicherung von Zeichenketten ist jedoch möglich.

Der Zabbix Server ist für den Empfang und die Weiterverarbeitung der eingehenden Daten verantwortlich. Zur Speicherung der eingehenden Daten können unterschiedliche Datenbanksysteme, wie z. B. DB2 oder MySQL, eingesetzt werden. Die gespeicherten Messwerte können in Tabellen dargestellt oder in verschiedenen Diagrammtypen visualisiert werden. Zur Darstellung der Daten in einem kleinen Intervall ist eine Anpassung der Oberfläche sowie der Datenbankabfrage notwendig. Die Datenbankabfrage gruppiert die eingegangen Messwerte für ein bestimmtes Intervall und bildet den Mittelwert über diese Gruppen. Lediglich der Mittelwert sowie Minima und Maxima werden in den Graphen dargestellt. Die Handhabung der Graphen ist komfortabel gestaltet. Der Zeitbereich der darzustellenden Daten kann manuell oder durch markieren eines Bereiches innerhalb des Graphen gewählt werden.

Zur Installation eines Zabbix Servers muss dieser zuvor aus den Quellen übersetzt werden, da keine aktuellen Installationspakete angeboten werden. Kompilierte Versionen des Agenten sind für verschiedene Betriebssysteme verfügbar. Die Konfiguration des Servers sowie der Agenten erfolgt größtenteils über die Weboberfläche. Über diese wird festgelegt, welche Messwerte durch welche Quelle erfasst werden sollen. Lediglich eine Grundkonfiguration erfolgt über entsprechende Dateien des Servers bzw. Agenten. Darüber hinaus bietet das Web-Frontend mittels JavaScript Object Notation Remote Procedure Call (JSON-RPC) eine API zur Steuerung des Servers bzw. Datenabfrage. Die umfangreiche Dokumentation umfasst den Aufbau der API sowie alle Informationen zum Einrichten und Betreiben des Monitoring-Systems. Verschiedene Anleitungen sind in Deutsch[24] verfügbar und durch den Hersteller wird ein Support angeboten.

---

[23] Release 2.0.1 vom 27.06.2012

[24] Siehe: http://lab4.org/wiki/Kategorie:Zabbix

3.3.6 Nagios

Tabelle 13 Bewertung Nagios Monitoring-System

Nagios Core 3.4.1 & Nagiosgraph 1.4.4	Punkte	Bemerkung
1) Überwachung		
Server		
CPU-Auslastung	2/2	Als Plug-in verfügbar
Netzwerk-Auslastung	2/2	Als Plug-in verfügbar
Speicherbelegung	2/2	Als Plug-in verfügbar
Anwendung		
Passive Überwachung	3/3	Über Add-on „NSCA“ verfügbar
Verarbeitung Zeichenketten	2/3	Plug-in Anpassung notwendig
Auswertung Logdateien	2/2	Als Plug-in verfügbar
Netzwerk		
ICMP Echo-Nachricht	1/1	Als Plug-in verfügbar
SNMP	2/2	Als Plug-in verfügbar
2) Analyse		
Darstellungsform		
Tabellen	1/3	Erstellung einer Ansicht notwendig
Graphen	3/3	
Übersichten	2/2	
Verarbeitung		
Sekündliche Skaleneinteilung	2/3	Anpassung notwendig
Grenzwert-Trigger	1/2	Plug-in muss Trigger unterstützen
3) Weiteres		
Technische Merkmale		
Konfiguration über Web-Frontend	1/2	Oberfläche von Dritten notwendig
Native Hostanwendungen	2/2	Über Add-on „NRPE“ verfügbar
Agenten-Manager Architektur	2/2	
Externe Schnittstellen	1/1	Mehrere Schnittstellen verfügbar
Nicht-technische Merkmale		
Handhabung Webfrontend/Graphen	2/2	Zusätzliches Add-on notwendig
Dokumentation/FAQ/Anleitungen	2/2	Umfangreiche Dokumentation
Aktive Gemeinschaft/Support	2/2	Bietet kommerziellen Support
Aufwand Installation/Wartung	0/2	Viele Plug-ins & Add-ons benötigt
Summe	37/45	

Nagios ist eine Monitoring-Software, die weite Verbreitung bei der Überwachung von Netzwerken gefunden hat. Nagios besteht aus einem Kernpaket, dem sogenannten Nagios Core[25], sowie aus zahlreichen Plug-ins und Add-ons. Das Kernpaket ist nicht in der Lage, Messwerte von einem Gerät oder einer Anwendung aufzunehmen. Diese Aufgabe wird von den zahlreichen Plug-ins übernommen, welche beispielsweise die Speicherbelegung oder CPU- und Netzwerkauslastung messen und die aufgenommenen Werte an den Kernprozess zur weiteren Verarbeitung senden. Weitere Plug-ins sind für die Überwachung von Netzwerkgeräten per SNMP oder von Logdateien vorhanden. Um die Messwerte von einem entfernten Rechner aufzunehmen, ist ein zusätzlicher Agent notwendig. Dieser steht als Add-on für verschiedene Betriebssysteme zur Verfügung und arbeitet nach dem Prinzip der aktiven Überwachung. Ein weiteres Add-on namens „Nagios Service Check Acceptor“ (NSCA) ist für die passive Überwachung von Linux und Unix basierten Rechnern notwendig.

Das Nagios Kernpaket bietet im Auslieferungszustand vergleichsweise wenige Visualisierungsmöglichkeiten. Hierzu existieren zahlreiche Weboberflächen als Erweiterung, die diesen Umstand aufheben. Für die Bewertung wird das populäre Add-on namens „Nagiosgraph“ in der aktuellen Version[26] verwendet. Dieses speichert die eingehenden Messwerte in einer Round-Robin Datenbank und erstellt aus den Werten entsprechende Graphen mithilfe des bekannten RRD Tools. Da diese Anwendung auch von anderen Überwachungssystemen wie z. B. Ganglia oder Cacti verwendet wird, stehen die gleichen Diagrammtypen zur Verfügung, die für eine Darstellung der Daten in einem kleinen Intervall angepasst werden müssen. Eine Ansicht der gespeicherten Daten in einer Tabelle muss gegebenenfalls ergänzt werden. Um die Messwerte auf das Erreichen eines bestimmten Grenzwertes zu überprüfen, muss diese Option durch das entsprechende Plug-in vorgesehen sein.

Nagios kann über passende Installationspakete oder aus den Quellen installiert werden. Neben der Installation des Kernpaketes ist es notwendig, alle benötigten Plug-ins einzeln zu installieren und einzurichten. Die Konfiguration von Nagios erfolgt ausschließlich über Dateien. Jedoch werden verschiedene Weboberflächen als Add-on bereitgestellt, die bei der Konfiguration unterstützen. Die Installation und Wartung des Überwachungssystems gestaltet sich je nach Anzahl der verwendeten Komponenten aufwendig. Die Dokumentation des Kernpaketes sowie der bereitgestellten Add-ons ist umfangreich. Durch das betreuende Unternehmen wird ein kommerzieller Support angeboten.

---

[25] Release 3.4.1 vom 11.05.2012

[26] Release 1.4.4 vom 15.01.2011

3.4 Fazit

Tabelle 14 Zusammenfassung der bewerteten Monitoring-Systeme

Zusammenfassung	Ganglia	Munin	Cacti	Zabbix	Nagios
Version	Core 3.4.0 Web 3.5.2	2.0.2	0.8.8a	2.0.1	Nagios Core 3.4.1 Nagiosgraph 1.4.4
Release-Datum	16.07.12	29.06.12	29.04.12	27.06.12	11.05.12
1) Überwachung	**13/17**	**11/17**	**12/17**	**15/17**	**16/17**
Server	6/6	6/6	6/6	6/6	6/6
Anwendung	6/8	2/8	3/8	6/8	7/8
Netzwerk	1/3	3/3	3/3	3/3	3/3
2) Analyse	**11/13**	**9/13**	**11/13**	**12/13**	**9/13**
Darstellungsform	7/8	6/8	6/8	8/8	6/8
Verarbeitung	4/5	3/5	5/5	4/5	3/5
3) Weiteres	**10/15**	**11/15**	**13/15**	**14/15**	**12/15**
Technische Merkmale	5/7	5/7	6/7	7/7	6/7
Nicht-technische Merkmale	5/8	6/8	7/8	7/8	6/8
Summe	**34/45**	**31/45**	**36/45**	**41/45**	**37/45**
Note	**2,1**	**2,4**	**1,9**	**1,4**	**1,8**

Die Zusammenfassung zeigt, dass sich die bewerteten Überwachungssysteme teils stark voneinander abgrenzen. In der Kategorie der Überwachungskriterien sind deutliche Einflüsse der unterschiedlichen Anwendungsbereiche erkennbar. Während sich die Entwickler von Ganglia auf die Überwachung von Servern, Clustern und den darauf laufenden Anwendungen konzentrierten, steht bei Munin und Cacti die Überwachung von unterschiedlichen Netzwerkkomponenten und Servern im Vordergrund. Zabbix und Nagios sind in diesem Bereich besonders flexibel und bieten ein großes Repertoire an unterschiedlichen Überwachungsfähigkeiten. Bei allen Systemen zeigt sich, dass die Verarbeitung von Zeichenketten durch eigene Erweiterungen implementiert bzw. bestehende Plug-ins für die geplante Verwendung angepasst werden müssen. Die Speicherung von eingehenden Zeichenketten wird lediglich von Zabbix unterstützt, da das eingesetzte Datenbankmanagementsystem RRD Tool für Round-Robin Datenbanken der anderen Überwachungssysteme nicht in der Lage ist, Zeichenketten zu verarbeiten.

In der Bewertung der Analysekriterien spiegeln sich weitere Eigenschaften des eingesetzten Datenbankmanagementsystems RRD Tool wider, da beispielsweise die Erstellung der Graphen von dieser Anwendung übernommen werden. Je nach Gebrauch, der von RRD Tool bereitgestellten Funktionen wie Schwellwertschalter, und Implementierung von eigenen Erweiterungen ergeben sich die jeweiligen Unterschiede unter diesen Überwachungssystemen. Zabbix sticht in dieser Kategorie hervor, da es als einzige Anwendung die eingegangen Daten in einer Tabelle wiedergibt. Ganglia ist lediglich in der Lage, die gespeicherten Daten über eine CSV-Datei zu exportieren.

In den weiteren Kriterien spiegeln sich unterschiedliche Implementierungs- und Designentscheidungen wider. So verzichtet beispielsweise Ganglia auf die Möglichkeit der Konfiguration über eine Weboberfläche. Auch die Unterstützung der Gemeinschaft rund um die Überwachungssysteme zeigt sich in den Punkteverteilungen. So existieren bei Nagios verschiedene Projekte, die Weboberflächen zur leichteren Konfiguration und Performanceanalyse entwickeln. Die Qualität der Dokumentation und die Unterstützung durch die Entwickler bzw. die Gemeinschaft sind bei Nagios und Zabbix am größten. Hinter beiden Projekten stehen Unternehmen, welche technische Unterstützung und Consulting-Dienstleistungen anbieten.

Zabbix geht aus der Bewertung als klarer Favorit für die spezielle Überwachung der Videoprozesskette hervor. Die Software bietet die flexibelsten Möglichkeiten der Überwachung sowie zur Verarbeitung und Visualisierung der eingegangenen Messwerte. Die Dokumentation und die technische Unterstützung sind im Vergleich zu anderen Systemen hervorragend.

4 Integration

4.1 Aufbau der Monitoring-Software

Aus dem vorherigen Abschnitt 3.4 ist bekannt, dass Zabbix als Favorit aus den bewerteten Überwachungssystemen hervorgeht. In diesem Abschnitt wird erläutert, wie Zabbix als Monitoring-Software grundsätzlich aufgebaut ist und auf welchem Wege die Überwachung eingerichtet wird und verläuft. Der Umfang ist auf Themen beschränkt, die zum grundlegenden Verständnis beitragen.

4.1.1 Zabbix Prozesse

Die Zabbix Monitoring-Software besteht aus mehreren Komponenten, die auf verschiedenen Systemen verteilt sein können und als Instanz eines Agenten bzw. Manager zu verstehen sind. Abbildung 20 zeigt eine mögliche Verteilung der Komponenten.

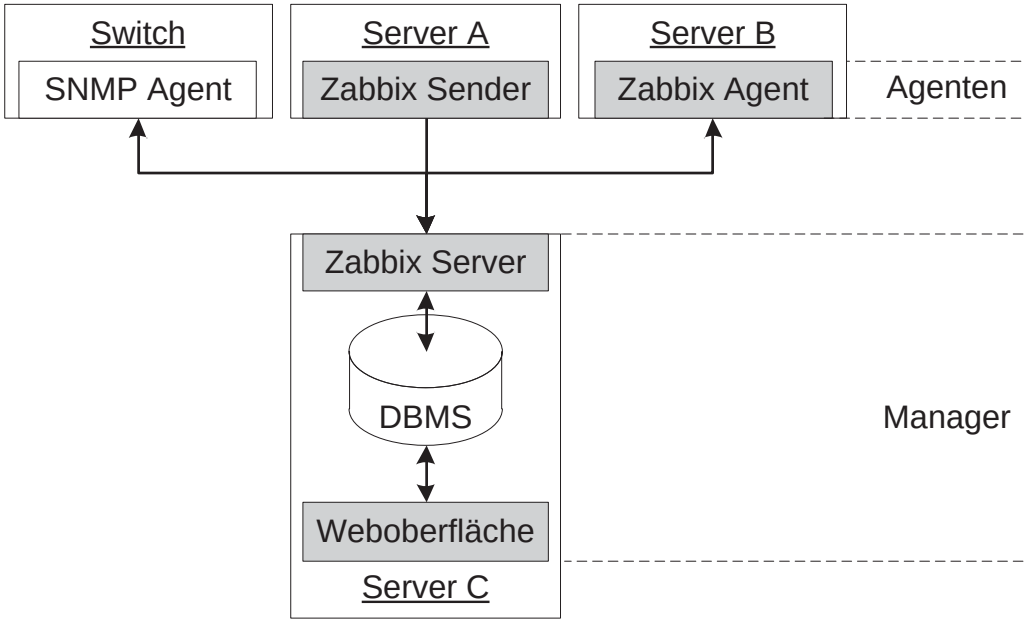

Abbildung 20 Monitoring Komponenten

Die Basis von Zabbix bilden die in C geschriebenen Anwendungen sowie die mittels PHP und Javascript erstellte Weboberfläche. Der Monitoring-Manager besteht aus dem Zabbix Server, der Weboberfläche sowie der angeschlossenen Datenbank. Hierbei unterstützt Zabbix unterschiedliche Datenbankmanagementsysteme, wie z. B. MySQL, PostgreSQL, SQLite, Oracle oder IBM DB2. Der Zabbix Server bezieht seine Aufgaben aus der Datenbank, welche zuvor über die Weboberfläche eingerichtet wurde, und speichert die erhaltenen Werte in dieser ab. Die Werte können von verschiedenen Quellen stammen. Als Agenten stehen zwei Anwendungen zur Verfügung, der Zabbix Agent sowie der

Zabbix Sender. Der Zabbix Sender kommuniziert mit dem Zabbix Server unidirektional und ermöglicht so nur eine passive Host-Überwachung, während der Zabbix Agent sowohl eine passive als auch aktive Überwachung ermöglicht. Der Zabbix Sender ist eine einfache Kommandozeilenanwendung zum Versenden von Leistungsdaten, unter Angabe des Host- und Item-Namen, an den Server. Der Zabbix Agent ist eine leistungsfähigere Anwendung mit einer Vielzahl von integrierten Metriken, die durch den Zabbix Server abgefragt werden können. Mithilfe von benutzerdefinierten Parametern kann der Zabbix Agent durch Skripte oder ausführbare Dateien erweitert werden. Des Weiteren werden verschiedene SNMP Agenten unterstützt, die nicht Teil der Monitoring-Software sind, sondern vom Gerätehersteller implementiert werden.

4.1.2 Objekte der Weboberfläche

Die Weboberfläche wird sowohl für die Konfiguration der Überwachung als auch für die Analyse der eingehenden Daten herangezogen. Hierzu werden über die Weboberfläche verschiedene Objekte definiert, die diese Aufgaben im Zusammenspiel erfüllen. Abbildung 21 zeigt ein beispielhaftes Objektdiagramm, in dem verschiedene Objekte der Überwachung und Analyse in ihrem Zusammenhang dargestellt sind.

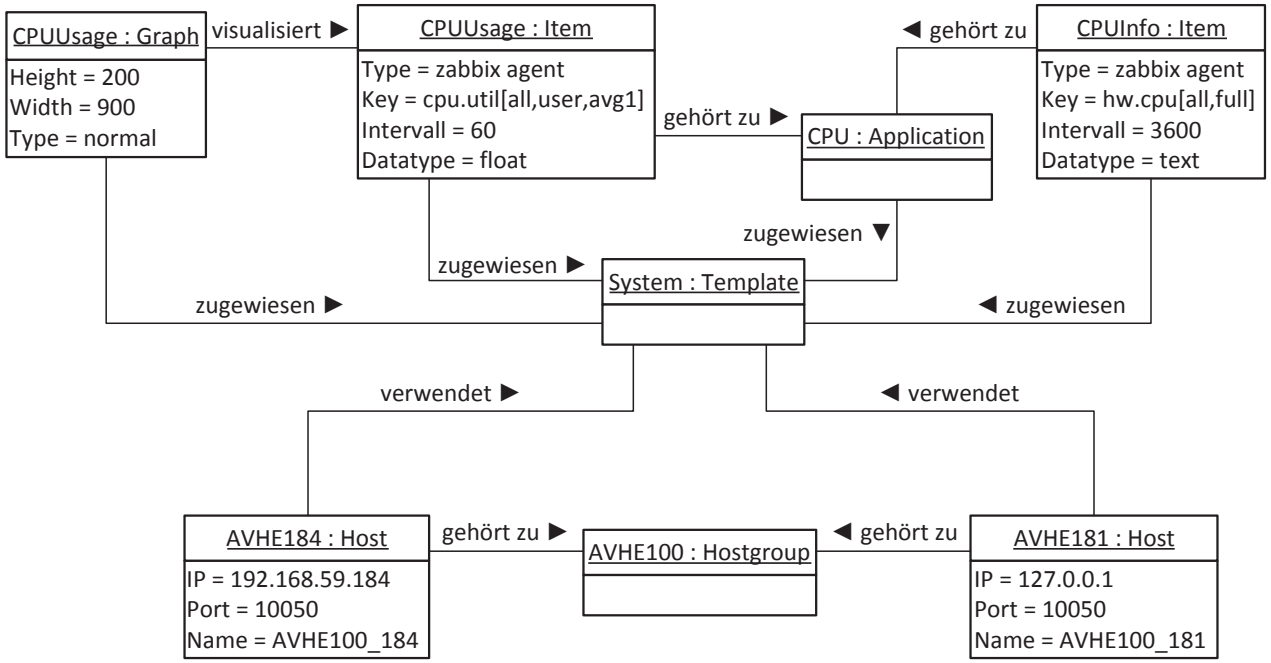

Abbildung 21 Objektdiagramm der Weboberfläche

Host

Alle physischen und virtuellen Maschinen und Geräte, die in einem Netzwerk mittels IP-Adresse oder DNS-Name ansprechbar sind, werden von Zabbix als Host betrachtet. Es ist dabei unerheblich, ob es sich um einen Server, Router, Netzwerkspeicher oder ein

ähnliches Gerät handelt, da in diesem Zusammenhang nicht der Typ des Gerätes sondern die Internetschnittstelle im Fokus steht.

Host-Gruppen

Eingerichtete Hosts können zu logischen Gruppen zusammengefasst werden, um die Übersicht bei einer hohen Anzahl an zu überwachenden Hosts behalten zu können.

Items

Items beschreiben die eigentlich zu messende Größe. Sie sind immer an einen jeweiligen Host gebunden. Einem Host können dabei beliebig viele Items zugeordnet werden. Der Zabbix Server hat nach Aktivierung des Items die Aufgabe, den Wert des Items im eingestellten Intervall auf dem zugeordneten Host zu messen.

Applikationen

Ähnlich den Host-Gruppen können Items, die verschiedene Aspekte eines Dienstes oder einer Anwendung beschreiben, in sogenannten Applikationen gruppiert werden.

Trigger

Trigger sind logische Ausdrücke, mit deren Hilfe die gemessenen Werte der Items mit einem definierten Schwellwert oder Werten anderer Items verglichen werden können. Wird ein solcher Schwellwert überschritten, ändert der Trigger seinen internen Status und kann als Auslöser einer bestimmten Aktion, z. B. dem Versand einer E-Mail, dienen. Trigger sind ein wichtiges Werkzeug zur Erkennung von Abweichungen des Normalzustandes.

Graphen

Alle eingegangenen Messwerte, die als Zahlenwerte gespeichert werden, können ohne weitere Konfiguration innerhalb der einfachen Graphen dargestellt werden. Neben den einfachen Graphen bietet Zabbix die Möglichkeit, benutzerdefinierte Graphen zu erstellen, die beispielsweise in der Lage sind, mehrere Werte in einem Graphen darzustellen.

Template

Ein Template beschreibt einen Satz von Entitäten wie z. B. Items, Trigger, Graphen, die auf einen oder mehrere Hosts angewendet werden. Wird einem Template ein Item zugewiesen, so werden die entsprechenden Daten von allen Hosts gesammelt, die dieses Template verwenden.

4.1.3 Prinzip der Konfiguration und Überwachung

Die Konfiguration der Anwendungen erfolgt über die entsprechenden Konfigurationsdateien. Über diese Dateien werden wesentliche Parameter der Kommunikation zwischen Manager und Agenten sowie Parameter zu Funktionseigenschaften der Anwendung gesetzt. In der Datei für den Zabbix Agenten wird so unter anderem die Server IP-Adresse, die benutzerdefinierten Parameter und der Hostname eingestellt. Der Zabbix Sender kann auf die Konfiguration des Agenten zurückgreifen, um die Server IP-Adresse und den Hostnamen zu ermitteln. Der Server speichert die empfangenen Daten dann unter dem vom Sender mitgelieferten Hostnamen ab. Weitere Details können der Dokumentation[27] entnommen werden.

Die Konfiguration der eigentlichen Überwachung erfolgt über die Weboberfläche. Im ersten Schritt wird ein zu überwachender Host eingerichtet. Abbildung 22 zeigt das entsprechende Konfigurationsmenü. Über dieses Menü erfolgen die Festlegung der IP-Adresse und des Interface-Ports sowie die Zuordnung von Templates und Gruppen. Hinter dem Host Interface verbirgt sich der Typ des eingesetzten Agenten, beispielsweise steht das einfache Agenten-Interface für einen installierten Zabbix Agenten auf dem Host.

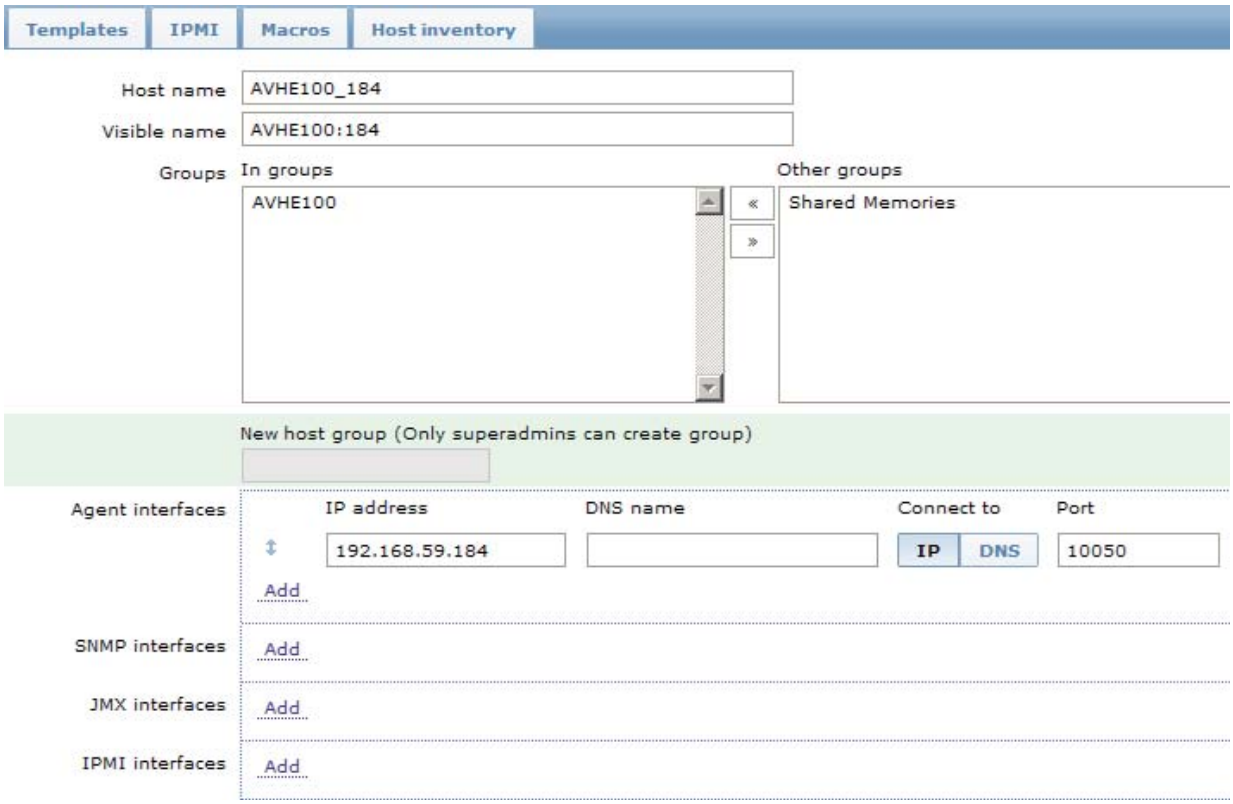

Abbildung 22 Einrichtung eines Hosts

[27] Siehe: http://www.zabbix.com/documentation/2.0/manual/appendix/config

Im nächsten Schritt wird für einen Host bzw. ein Template ein Item für die zu überwachende Ressource angelegt. Ausgewählte Parameter zur Einrichtung eines Items sind in Abbildung 23 abgebildet. Auf die wesentlichen Parameter wird an dieser Stelle kurz eingegangen. Für eine ausführliche Beschreibung wird jedoch auf die Zabbix Dokumentation[28] verwiesen. Der Typ des Items spezifiziert die Quelle der Messdaten. Zur Auswahl stehen unter anderem der Zabbix Agent, Agenten der SNMP Protokolle sowie der Zabbix Trapper. Dieser stellt den Empfänger für eingehende Messdaten dar, die von einem Zabbix Sender an den Server gesendet werden. Der Schlüssel beschreibt die eigentliche Messgröße. Der Zabbix Agent unterstützt eine Vielzahl von Schlüsseln wie z. B. zur Erfassung der CPU-Auslastung für alle Benutzerprozesse (dargestellt in Abbildung 23), Arbeitsspeicherbelegung oder Netzwerkauslastung. Der Parametername eines Schlüssels kann unter Angabe der Parameternummer mittels Dollarzeichens, z. B. $2, im Item-Namen verwendet werden. Unter dem Informationstyp ist der eigentliche Datentyp der Messgröße zu verstehen. Zabbix unterstützt hierzu vorzeichenlose ganze Zahlen (64bit), Gleitkommazahlen (32bit) sowie einzelne Zeichen und Zeichenketten. Wird als Datentyp eine vorzeichenlose ganze Zahl gewählt, kann zusätzlich das Stellenwertsystem, wie z. B. hexadezimal, ausgewählt werden. Zabbix bezeichnet dies irrtümlich als Datentyp!

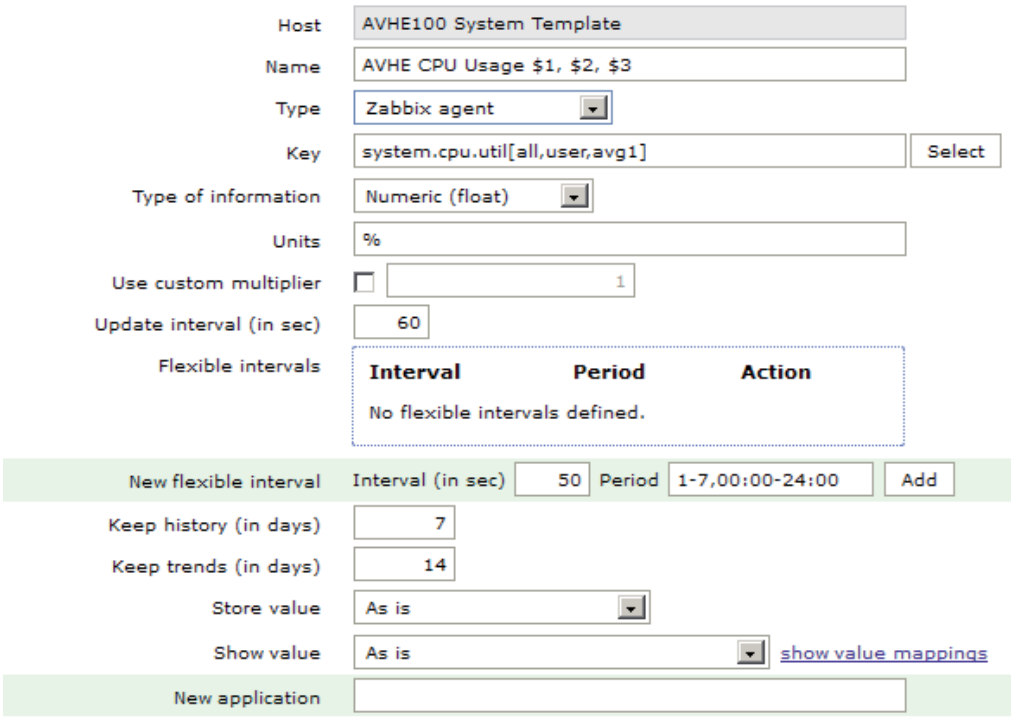

Abbildung 23 Einrichtung eines Items

[28] Siehe: http://www.zabbix.com/documentation/2.0/manual/config/items

Die erfassten Werte können mittels Multiplikator skaliert werden. Des Weiteren ist die Speicherung der Differenz des aktuellen Wertes zum vorhergehenden Wert möglich, um so z. B. den Datendurchsatz in Bytes pro Sekunde zu erhalten. Liefert das Item eine definierte Anzahl an Werten zurück, wie z. B. 0 oder 1, können diesen Werten bestimmte Textketten zugeordnet werden (Mapping), um die Lesbarkeit der Rückgaben zu erhöhen. Sind mehrere Items einem bestimmten Dienst oder Anwendung zugeordnet, können diese innerhalb einer Applikation gruppiert werden.

Nach erfolgreicher Einrichtung eines Hosts und der Zuordnung von Items, kann die Überwachung des Hosts gestartet werden. Hierzu muss der Status des Hosts auf „überwacht“ geändert und das Item der zu überwachenden Ressource aktiviert werden. Anschließend erhält der Zabbix Server aus der Datenbankabfrage den Auftrag, das Item im angegebenen Intervall vom zugeordneten Host abzufragen und die erhaltenen Messwerte in der Datenbank abzulegen. Ist als Interface der Zabbix Agent eingerichtet, wird dieser die Anfrage entgegennehmen und den entsprechenden Messwert des Schlüssels aufnehmen. Steht hinter dem Schlüssel ein benutzerdefinierter Parameter, beispielsweise ein ausführbares Skript, welches einen bestimmten Wert generiert, muss dieses innerhalb der Zeitbeschränkung den Wert liefern. Abbildung 24 zeigt den beispielhaften Ablauf innerhalb eines Sequenzdiagrammes.

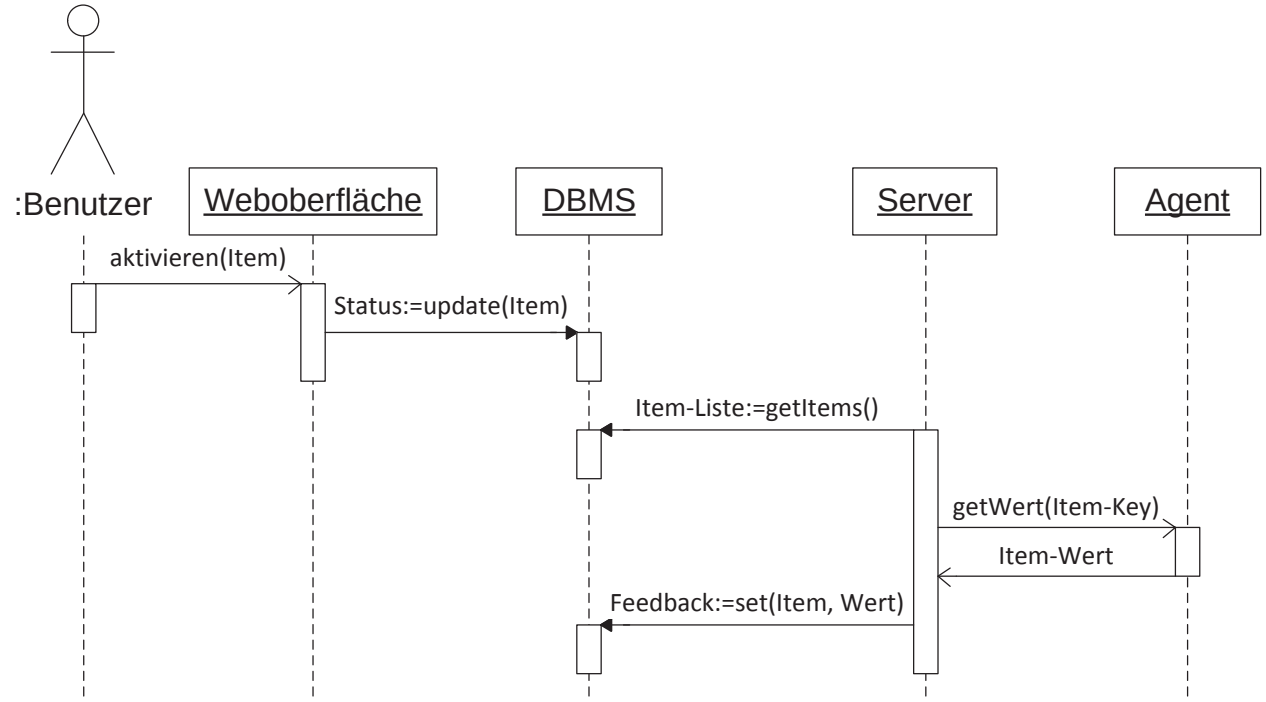

Abbildung 24 Sequenzdiagramm der Hostüberwachung

Nach Deaktivierung des Items oder Änderung des Host-Status wird die Überwachung der Ressource bzw. des gesamten Hosts gestoppt.

4.2 Entwurf

In den Anforderungen der Anwendungsüberwachung aus Abschnitt 3.2.1.2 ist festgehalten, dass die Konsolenanwendung „Spucat“ zu verwenden ist, um die Informationen des Shared Memory Headers sowie die Metadaten der enthaltenen Slots zu extrahieren. Diese Anwendung ist dabei in der Lage, sich als lesender Prozess an jeden Shared Memory der Videoprozesskette anzuhängen und die enthaltenen Informationen auf der Konsole als aufbereitete Textkette auszugeben. Dabei werden die Nutzdaten unter Linux auf der Standardausgabe (stdout) und die Headerinformationen sowie die Metadaten der Slots auf der Standardfehlerausgabe (stderr) wiedergegeben. Für die Überwachung der Prozesskette sind dabei nur die Ausgaben auf der Standard-fehlerausgabe relevant. Diese Textketten werden von einem zusätzlichen Programm, auch „Spucat Bridge“ genannt, aufbereitet, um die Informationen mittels des Zabbix Senders an den Überwachungsserver senden zu können. Dieses Konzept wird in Abbildung 25 dargestellt und im Laufe dieses Abschnitts detailliert erläutert.

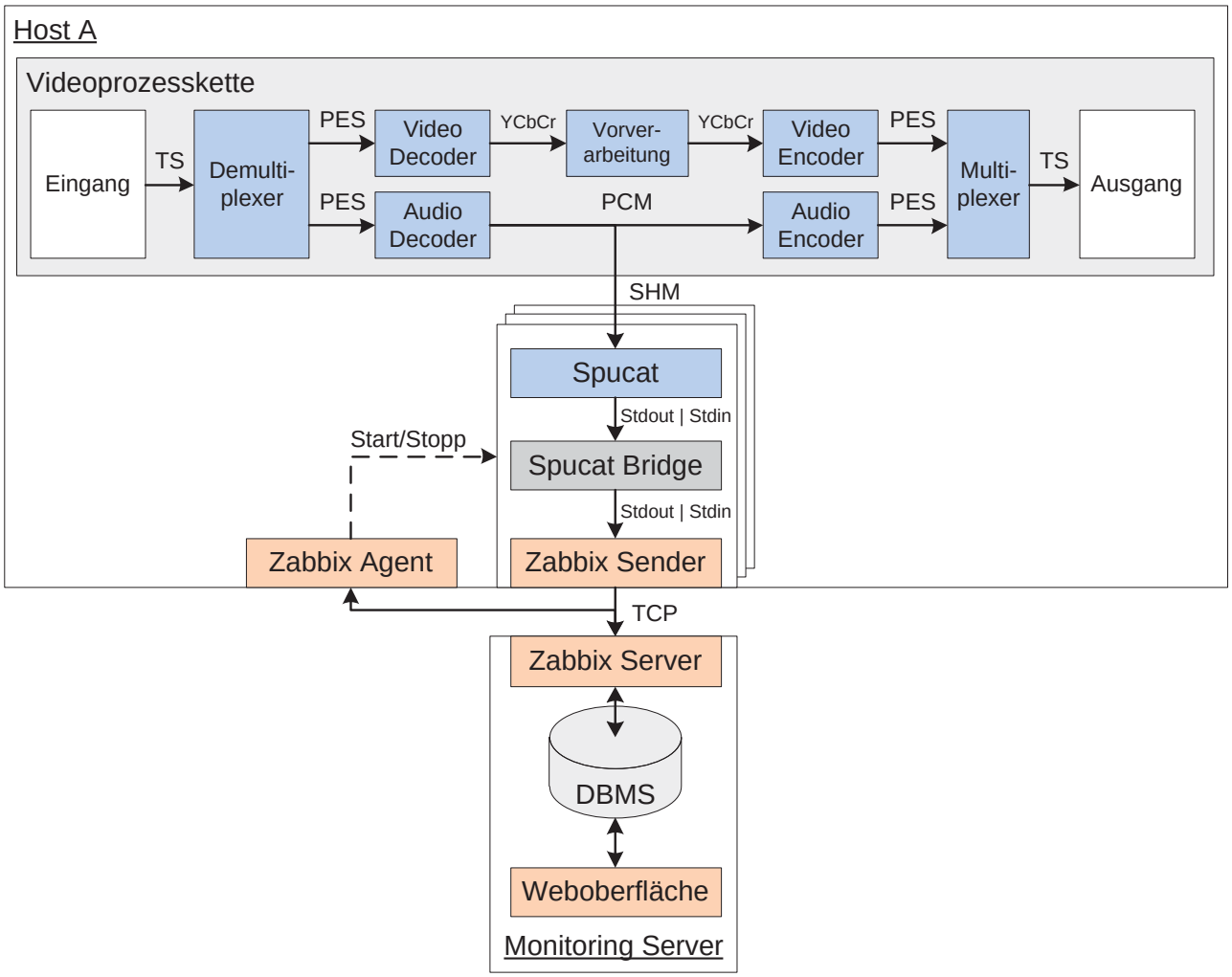

Abbildung 25 Entwurf der Überwachungskette

4.2.1 Datenübertragung zum Monitoring Server

Aus dem Abschnitt 3.1.2.2 ist bekannt, dass Spucat als lesender Prozess eine Update-Mitteilung nach jedem abgeschlossenen Schreibvorgang des schreibenden Prozesses erhält. Nach Erhalt dieser Nachricht können die geschriebenen Daten ausgelesen und weiterverarbeitet werden. Je nachdem welche Daten der gemeinsame Speicherbereich enthält, können die Update-Mitteilungen in einem Intervall von 20ms auftreten, wie zum Beispiel bei einem Videosignal mit 50 Bildern pro Sekunde. Um die Daten am Ende der Verarbeitung an den Monitoring Server zu senden, wird der Zabbix-Sender verwendet. Denkbar ist auch die Verwendung des Zabbix Agenten, jedoch sprechen mehrere Gründen gegen dessen Einsatz. Zum einen muss ein Mechanismus geschaffen werden, über welchen dem Agenten die Ausgangsdaten von Spucat bereitgestellt werden, im einfachsten Fall kann dies über eine Text-Datei geschehen. Zum anderen muss der Agent soweit erweitert werden, dass er in der Lage ist, mehrere verschiedene Item-Werte von unterschiedlichen Zeitpunkten an den Server zu senden. Das kleinste Intervall, in dem der Agent angesprochen werden kann, beträgt 1 Sekunde. Der Agent muss folglich im obigen Beispiel allein von einem Item 50 unterschiedliche Werte in dieser Sekunde an den Server senden. Der Sender hingegen ist im Stande, mehrere Werte von gleichen und verschiedenen Items an den Server zu senden, da hier dem Server explizit mitgeteilt wird, zu welchem Item der gesendete Wert gehört und zu welchem Zeitpunkt dieser aufgenommen wurde. Der Zabbix Sender liest hierzu die zu sendenden Werte aus einem angegebenen Textdokument oder von der Standardeingabe ein. Jede Zeile des Textdokumentes enthält dabei, durch Leerzeichen getrennt, den Hostnamen, wie er über die Weboberfläche angegeben wurde, den Schlüssel des Items, optional einen Zeitstempel im Unix-konformen Format sowie den eigentlichen Messwert:

```
Zeile = Hostname ‘ ‘ Schlüssel ‘ ‘ [Zeitstempel ‘ ‘] Wert ‘\n‘
```

Im gleichen Format können dem Sender die Daten über die Standardeingabe übergeben werden. Der Zabbix Sender bietet eine Echtzeit-Option, die bewirkt, dass die eingehenden Werte so früh wie möglich an den Sender gesendet werden. Um den Overhead der Datenübertragung mittels Transportprotokolls zum Server so gering wie möglich zu halten, werden eingehende Messwerte aggregiert, die in einem zeitlichen Abstand von 0,2 s erfasst wurden. Diese Messwerte werden anschließend jede Sekunde an den Monitoring Server gesendet. Für weitere Details zur Verwendung des Senders wird auf das Onlinehandbuch[29] verwiesen.

[29] Siehe: http://www.zabbix.com/documentation/2.0/manpages/zabbix_sender

4.2.2 Funktion der Brückenanwendung

Da sich das Format der Spucat-Ausgaben grundsätzlich von dem Format der Eingangsdaten des Senders unterscheidet, ist eine zusätzliche Anwendung notwendig, die die Ausgaben in das erforderliche Format konvertiert. Diese Aufgabe übernimmt die sogenannte „Spucat Bridge“. Diese bekommt über die Standardeingabe die Ausgangsdaten von Spucat weitergeleitet, um die Informationen aus den Ausgaben zu extrahieren sowie den entsprechenden Schlüssel hinzuzufügen. Als Hostnamen wird lediglich ein Bindestrich angegeben, da dieser vom Sender aus der Konfigurationsdatei des Agenten gelesen oder als Argument übergeben wird. Die entsprechenden Item-Schlüssel (siehe Anlage 2 bis Anlage 6) werden innerhalb der Spucat Bridge für eine erste Implementierung statisch im Quelltext verankert und über eine Headerdatei bekannt gemacht. Der Hostname sowie die Items mit den entsprechenden Schlüsseln müssen zuvor über die Weboberfläche eingerichtet und manuell übertragen werden. Abbildung 26 zeigt gekürzte beispielhafte Ausgaben mit den entsprechenden Eingabedaten.

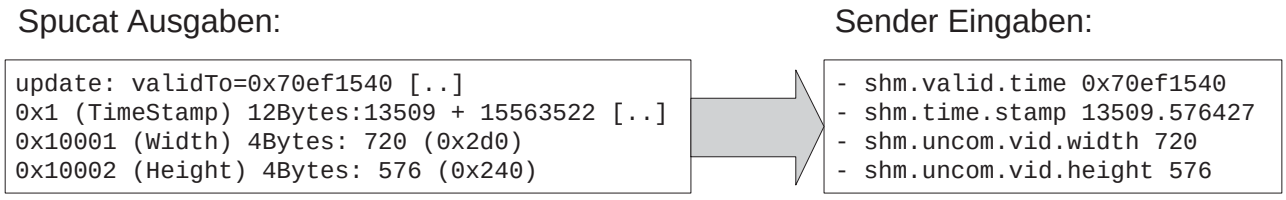

Abbildung 26 Beispielkonvertierung der Spucat Ausgaben

Die Spucat Ausgaben untergliedern sich dabei in zwei Gruppen. Die Gruppen stimmen mit den Headerinformationen des Shared Memories bzw. des Slots sowie den Metadaten überein (siehe Abschnitt 3.1.2.1). Die Ausgabezeile der Headerinformationen des Shared Memories und des Slots wird mit der Zeichenkette „update:“ eingeleitet und enthält die Werte des virtuellen Zeitstempels, des gültigen Zeitstempels, des Typs sowie des Daten Offsets. Die einzelnen Werte beginnen mit einer bestimmten Zeichenkette, wie z. B. „validTo=“ für den gültigen Zeitstempels und sind durch Leerzeichen voneinander getrennt. Mithilfe dieser Informationen können die Werte der Update-Zeile gesucht, extrahiert und mit dem entsprechenden Item-Schlüssel versehen werden. Anhand des Hexadezimalen Präfixes „0x“ können die Metadaten des Slots erkannt werden, da die hexadezimale Zahl am Anfang jeder Zeile dem Marker der Meta-Information entspricht. Der maximal 4-Byte große Wert der Meta-Information steht am Ende jeder Zeile in runden Klammern eingeschlossen, zum Beispiel „(0x2d0)“ aus Abbildung 26. Zu beachten ist hierbei, dass Zeitstempel-Informationen nicht diesem Schema folgen, wie in Abbildung 27 zu sehen ist.

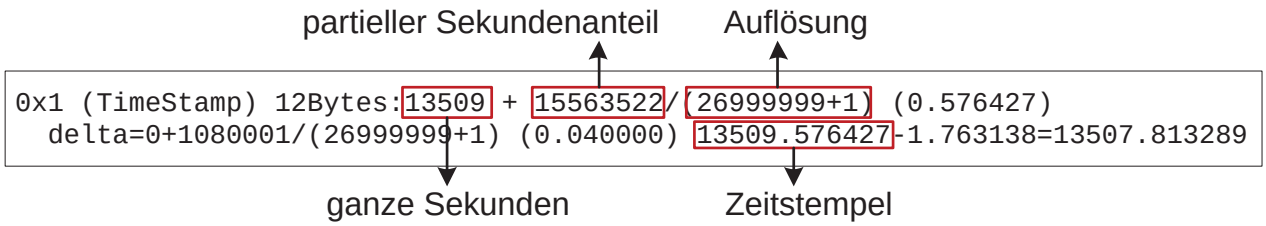

Abbildung 27 Spucat Ausgabe eines Zeitstempels

Die Ausgabe eines Zeitstempels enthält, neben den zu extrahierenden rationalen Sekunden, alle Bestandteile, um diesen Wert zu berechnen. Mit diesem Sonderfall ergeben sich somit drei Kategorien von möglichen Ausgaben, die durch jeweils eine Funktion innerhalb der Anwendung durchsucht werden, um die entsprechende Information zu extrahieren. Nach der Verarbeitung werden die Daten auf der Standardausgabe ausgegeben und mittels Pipe an den Sender weitergeleitet, der diese an den Monitoring Server sendet.

4.2.3 Aufgaben des Agenten

Auf dem zu überwachenden Host ist zusätzlich der Zabbix Agent installiert. Diesem kommen die beschriebenen Aufgaben aus Abschnitt 3.2.1.1 zu. Hierzu zählen unter anderem die Erfassung der CPU- und Netzwerkauslastung sowie die Belegung des Arbeitsspeichers. Zur Erfüllung dieser Aufgaben werden entsprechende Items über die Weboberfläche eingerichtet. Über spezielle Start- und Stopp-Items wird die Überwachung eines Shared Memories gestartet bzw. eine laufende Überwachung beendet. Hinter diesen Items verbirgt sich jeweils ein benutzerdefinierter Parameter mit einem Kommandozeilenbefehl. Durch Ausführung eines der beiden Befehle wird ein Skript gestartet, welches die Überwachungsprozesskette, bestehend aus Spucat, Spucat Bridge sowie dem Zabbix Sender, mit den notwendigen Parametern startet bzw. beendet. Als Start-Parameter wird dem Skript das zu überwachende Shared Memory übergeben. Das Skript gibt nach Ausführung einen entsprechenden Status über den Erfolg bzw. Misserfolg der Ausführung zurück. Dieser Status kann anschließend über die Weboberfläche eingesehen werden.

Im ersten Schritt der Implementierung des Überwachungsprozesses wird keine parallele Überwachung mehrerer Shared Memories unterstützt. Dieser Stand der Entwicklung ist die Grundlage dieser Arbeit. Die Konfiguration der Weboberfläche sowie die Skripte zum Starten und Beenden sind jedoch daraufhin ausgelegt, durch wenige Anpassungen eine Überwachung von mehreren Shared Memories zu ermöglichen. Hierzu wird dem Zabbix Sender der Hostname über ein Kommandozeilenargument übergeben, anstatt diesen aus der Konfigurationsdatei des Agenten zu beziehen. Der Hostname wird in diesem Fall über die Weboberfläche konfiguriert und als zweiter Parameter dem Startskript übergeben.

4.3 Implementierung

Im vorhergehenden Abschnitt 4.2 wurde erläutert, aus welchen Bestandteilen die Überwachungsprozesskette besteht, welche Aufgaben die einzelnen Komponenten zu erfüllen haben und wie die Kommunikation unter diesen Prozessen stattfindet. Dieser Abschnitt konzentriert sich auf die Umsetzung der notwendigen Konfigurationen und des eigentlichen Überwachungsablaufes.

4.3.1 Spucat Bridge

Abbildung 28 zeigt den Programmablaufplan der Brückenanwendung. Dabei werden die gelesenen Eingaben der Standardeingabe auf das Hexadezimale-Präfix bzw. die Update-Zeichenkette untersucht. Wenn eine entsprechende Zeichenkette gefunden wurde, werden die Funktionen zur Datenextraktion aufgerufen. Diese beinhalten die entsprechenden Formatierungsfunktionen zur Ausgabe der Daten auf der Standardausgabe.

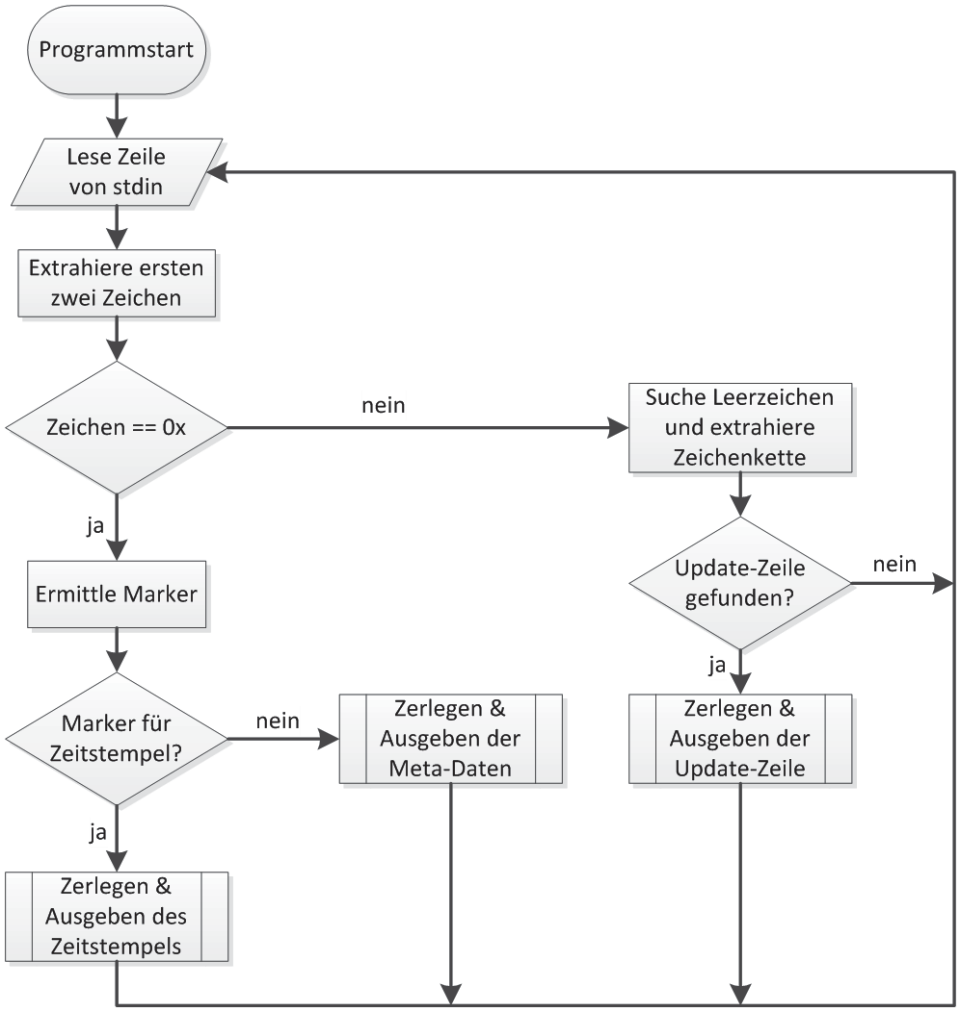

Abbildung 28 Programmlaufplan der Spucat Bridge

4.3.2 Eingerichtete Zabbix Konfiguration

4.3.2.1 SHM Start-Stop Template

Innerhalb dieses Templates sind mehrere Items eingerichtet, welche weniger für die Erfassung von Messwerten zuständig sind, sondern zur Steuerung der Shared Memory Überwachung verwendet werden.

Tabelle 15 Items des SHM Start-Stop Templates

Item-Name	Schlüssel	Erläuterung
SHM Accessible	vfs.file.exists [/dev/shm/{$SHM}]	Überprüft, ob die Datei unter dem Host-Makro {$SHM} unter dem angegeben Pfad existiert. Gibt 0 zurück wenn die Datei nicht existiert bzw. 1 wenn diese existiert. Über die angelegte Wertzuordnung „SHM Availability“ wird dem Wert 0 die Zeichenkette „Failed: SHM does not exist“ und dem Wert 1 „Ok: SHM exists“ zugeordnet
SHM Start	shm.start[{$SHM}]	Startet die Überwachung des unter dem Host-Makro {$SHM} angegebenen Shared Memories
SHM Stop	shm.stop	Stoppt die Überwachung

Über das Host-Makro „{$SHM}“ wird das zu überwachende Shared Memory in der Host-Konfiguration über die Weboberfläche festgelegt. Die Existenz dieses Shared Memories wird regelmäßig vom Agenten auf dem Server mittels des Items „SHM Accessible“ überprüft. Auf diese Weise können fehlerhafte Shared Memory Namen festgestellt werden. Unter den zuletzt eingegangenen Daten kann das Ergebnis der Überprüfung eingesehen werden.

Der Agent bekommt über das Item „SHM Start“ den Shared Memory Namen zum Start der Überwachung übergeben. Der Schlüssel dieses Items gehört zu einem benutzerdefinierten Parameter des Agenten, welcher in der Konfigurationsdatei des Agenten zu finden ist:

```
UserParameter=shm.start[*],sudo /etc/init.d/zabbix-sender start $1
```

Dem Item Schlüssel folgt, durch Komma getrennt, der Befehl, welcher durch den Agenten ausgeführt wird. Durch den Wildcard-Parameter „[*]“ wird dem Agenten mitgeteilt, die vom Server gesendeten Parameter an das Start-Stop-Skript „zabbix-sender“ weiterzuleiten. Dieses Skript startet bei Ausführung die Prozesskette zur Überwachung des angegebenen Shared Memories.

Der zweite benutzerdefinierte Parameter gehört zum Item „SHM Stop“:

```
UserParameter=shm.stop,sudo /etc/init.d/zabbix-sender stop
```

Über diesen Befehl wird die laufende Überwachung des Shared Memories beendet. Hierzu sind keine Parameter notwendig. Die zurückgegebenen Statusmeldungen über den Erfolg bzw. Misserfolg der Ausführung werden durch den Agenten an den Server weitergeleitet. Diese sind über die Weboberfläche unter den zuletzt eingegangenen Daten einsehbar.

4.3.2.2 Hostgruppen und Hosts

Wie in Abbildung 29 zu sehen, beinhaltet die eingerichtete Zabbix Installation zwei eingerichtete Host-Gruppen mit den zugehörigen Templates und zugeordneten Hosts.

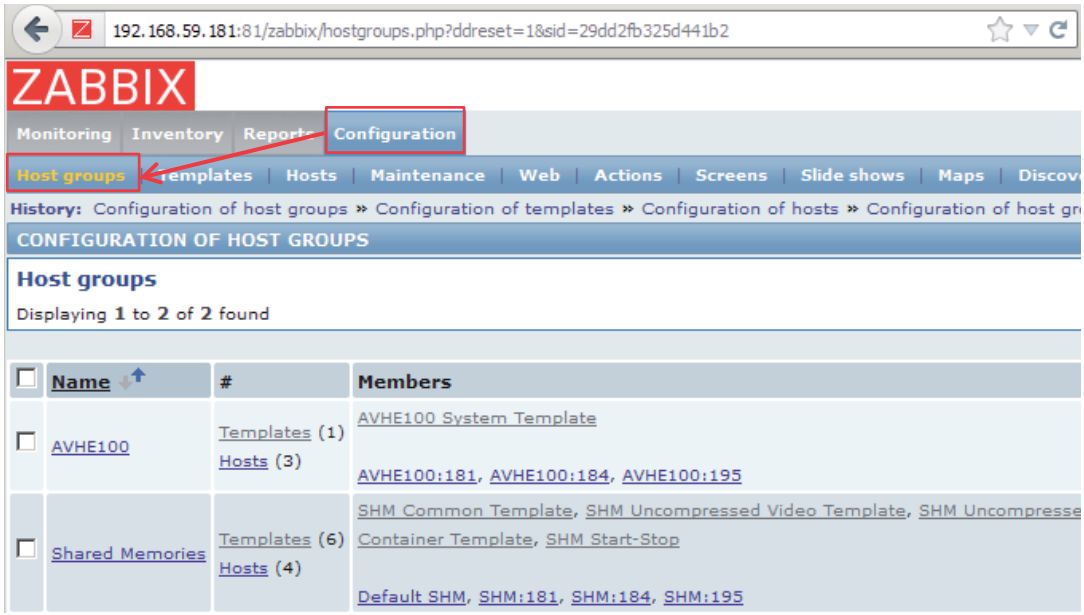

Abbildung 29 Eingerichtete Host-Gruppen

Die Host-Gruppe „AVHE100“ vereint alle physischen Server unter sich, d. h. ihr werden alle zu überwachenden Transcoding-Server zugeordnet. Dieser Gruppe ist lediglich das Template „AVHE100 System Template“ zugeordnet. Innerhalb dieses Templates sind verschiedene Items eingerichtet, die Informationen über die Serverhardware sammeln, wie beispielsweise die CPU-Auslastung (siehe Anlage 1). Die Werte dieses Templates werden vom Zabbix Agenten direkt auf dem Server gemessen.

Innerhalb der Host-Gruppe „Shared Memories“ werden alle überwachten Shared Memories auf den Transcoding-Servern zusammengefasst. Die Einrichtung von eigenen Hosts ist notwendig, da auf einem physischen Server mehrere Shared Memories innerhalb der Verarbeitungskette zur Interprozesskommunikation verwendet werden. Um zu einem späteren Zeitpunkt eine parallele Überwachung mehrerer Shared Memories zu ermöglichen, ist es von Anfang an zweckmäßig, eigene Hosts für die Shared Memories einzurichten, da der Zabbix-Sender die erfassten Werte immer an einen übergebenen Hostnamen und Item-Schlüssel bindet, unter welchem die erfassten Messwerte vom

Zabbix Server abgespeichert werden. Sollen mehrere Shared Memories überwacht werden, müssen Werte mit gleichem Item-Schlüssel getrennt abgespeichert werden, um eine Vermischung der Datensätze zu vermeiden. Diese Trennung erfolgt durch die Einrichtung von je einem Host pro überwachten Shared Memory. Der Host-Gruppe „Shared Memories“ sind alle Templates zugeordnet, die mit der Überwachung der Shared Memories in Verbindung stehen. Dazu gehören das:

- SHM Common Template (siehe Anlage 2)
- SHM Compressed Video Template (siehe Anlage 3)
- SHM Container Template (siehe Anlage 4)
- SHM Uncompressed Audio Template (siehe Anlage 5)
- SHM Uncompressed Video Template (siehe Anlage 6)
- SHM Start-Stop Template (siehe Abschnitt 4.3.2.1)

Wie in Abschnitt 4.3.2.1 beschrieben, benötigt das Start-Stopp-Skript den Shared Memory Namen, welcher überwacht werden soll. Unter dem Reiter „Macros“ im Host-Konfigurationsmenü der Weboberfläche ist das Host-Makro „{$SHM}“ für Hosts der Gruppe „Shared Memories“ definiert, welches den zu überwachenden Shared Memory Namen enthält. Zur Änderung des zu überwachenden Shared Memories ist der Inhalt der Makros zu ändern.

4.3.3 Überwachungsablauf

Im Abschnitt 4.3.2.1 wurde bereits das Start-Stopp-Template mit den dazugehörigen benutzerdefinierten Templates erläutert. Mittels Aktivierung des Start-Items wird der Agent auf dem zugehörigen Host durch den Server angewiesen, den Befehl des benutzerdefinierten Parameters auszuführen. Der Agent startet somit das Skript „zabbix-sender“ mit dem entsprechenden Parameter „start“. Zum Starten der Überwachung ist als zweiter Parameter die Angabe des Namens des zu überwachenden Shared Memories notwendig. Vor dem Start des Überwachungsprozesses prüft das Skript, ob bereits ein Prozess des Zabbix Senders ausgeführt wird:

```
if [ "zabbix_sender" = "ps -fC zabbix_sender | grep -o zabbix_sender" ]
```

Der Befehl „ps –fC“ listet den angegebenen Prozess „zabbix_sender“ aus allen laufenden Prozessen auf. Die Ausgaben werden an das Programm „grep“ weitergeleitet, welches durch die Option „-o“ die Ausgaben auf die Zeichenkette „zabbix_sender“ trimmt, wenn diese vorhanden ist. Wird ein laufender Prozess gefunden, wird die Ausführung nach Rückgabe der Statusmeldung „Already running: Zabbix Sender Service“ abgebrochen und keine weitere Überwachungsinstanz gestartet. Da der Zabbix Server die Ausführung des Skriptes im angegeben Intervall des Start-Items wiederholt anweist, ist die Überprüfung notwendig, um mehrfache Überwachungsinstanzen zu unterbinden. Eine einmalige

Abfrage durch den Server ist nicht möglich. An dieser Stelle wird sichergestellt, dass maximal eine Überwachungsinstanz gestartet werden und somit ein Shared Memory überwacht werden kann. Wenn kein laufender Prozess gefunden wurde, wird die Statusmeldung „Starting Zabbix Sender Service: $2“ (mit Angabe des zu überwachenden Shared Memories anstelle von „$2“) zurückgegeben und das Programm „start-stop-daemon“ gerufen, welches mithilfe eines weiteren Skriptes die Überwachungsprozesse startet. Hierzu wird durch das Skript „start_bridge“ der folgende Befehl gerufen:

```
$SPUCAT_PATH/spucat $SHM 2>&1 1>/dev/null | $BRIDGE_PATH/spucat_bridge \
| $SENDER_PATH/zabbix_sender $SENDER_OPT 1>/dev/null
```

Die Variablen „$SPUCAT_PATH“, „$BRIDGE_PATH” sowie “$SENDER_PATH” enthalten jeweils die Pfadangaben zu den einzelnen Programmen. Alle Optionen des Senders sind in der Variable „$SENDER_OPT“ festgehalten. Dazu gehören die Angabe der Konfigurationsdatei, die Echtzeitoption sowie die Angabe, dass die Daten von der Standardeingabe gelesen werden sollen. Der Name des zu überwachenden Shared Memories steht in der Variable „$SHM“. Um die durch Spucat gelesenen Headerinformationen des Shared Memories sowie die Metadaten an die Brückenanwendung weiterzuleiten, müssen diese zuvor mittels der Angabe „2>&1“ auf die Standardausgabe umgeleitet werden. Die ursprüngliche Standardausgabe des Senders sowie von Spucat werden verworfen.

Das Sequenzdiagramm aus Abbildung 30 veranschaulicht den vereinfachten Ablauf zum Starten der Datenerfassung von einem Shared Memory. Aus Platzgründen wird in dem Diagramm die Aktivierung des Items durch den Benutzer nicht dargestellt. Die vorangegangen Aktionen folgen dennoch dem in Abbildung 24 dargestellten Ablauf.

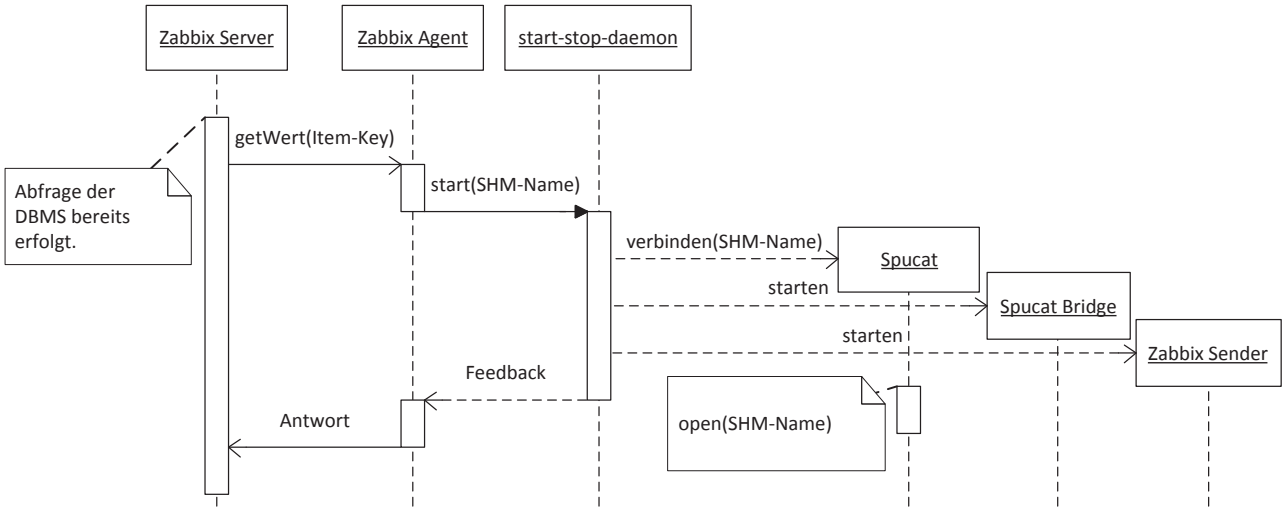

Abbildung 30 Sequenzdiagramm zum Starten der SHM Überwachung

Nachdem die Prozesse zur Datenerfassung gestartet wurden, liest Spucat die notwendigen Daten aus dem Header des Shared Memories, um sich bei dem schreibenden Prozess anzumelden. Hierzu gehören die Shared Memory ID, die IP-Adresse sowie der Port des Schreibers. Nach dem erfolgreichen Schreiben eines Slots versendet der schreibende Prozess eine Update-Nachricht an alle verbundenen lesenden Prozesse. Nach Erhalt dieser Nachricht, liest Spucat die Metadaten des zu überwachenden Shared Memories aus und gibt diese an die Brückenanwendung weiter. Diese wandelt die Daten in ein Zabbix-konformes Format, die mithilfe des Senders an den Server gesendet und von diesem gespeichert werden. Das Sequenzdiagramm aus Abbildung 31 zeigt den Ablauf detailliert.

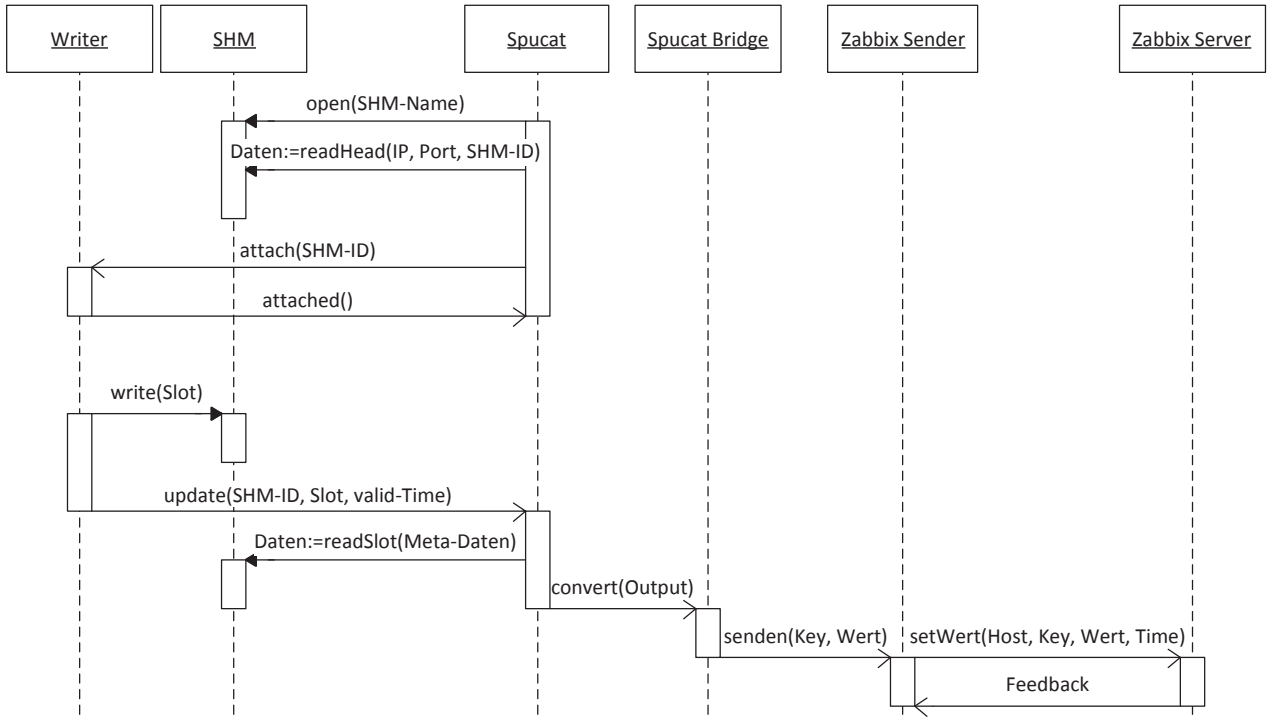

Abbildung 31 Sequenzdiagramm zur Datenerfassung

Vor Aktivierung des Stop-Items ist darauf zu achten, dass das Start-Item deaktiviert ist. Gleiches gilt bei Aktivierung des Start-Items. Zu keinem Zeitpunkt sollten beide Items gleichzeitig aktiv sein, da sonst die Überwachungsinstanz ständig gestartet und nach kurzer Zeit wieder beendet wird. In diesem Zustand kann keine kontinuierliche Datenerfassung durch die Überwachungsprozesse erfolgen.

Durch Aktivierung des Stop-Items über die Weboberfläche wird durch den Agenten das Skript „zabbix_sender“ mit dem Parameter „stop“ gerufen. Die einzelnen Prozesse werden mithilfe des Programmes „start-stop-daemon“ beendet und eine entsprechende Statusmeldung „Stopping Zabbix Sender Service“ wird an den Server zurückgegeben. Abbildung 32 zeigt den vereinfachten Ablauf zum Beenden des Überwachungsprozesses innerhalb eines Sequenzdiagrammes.

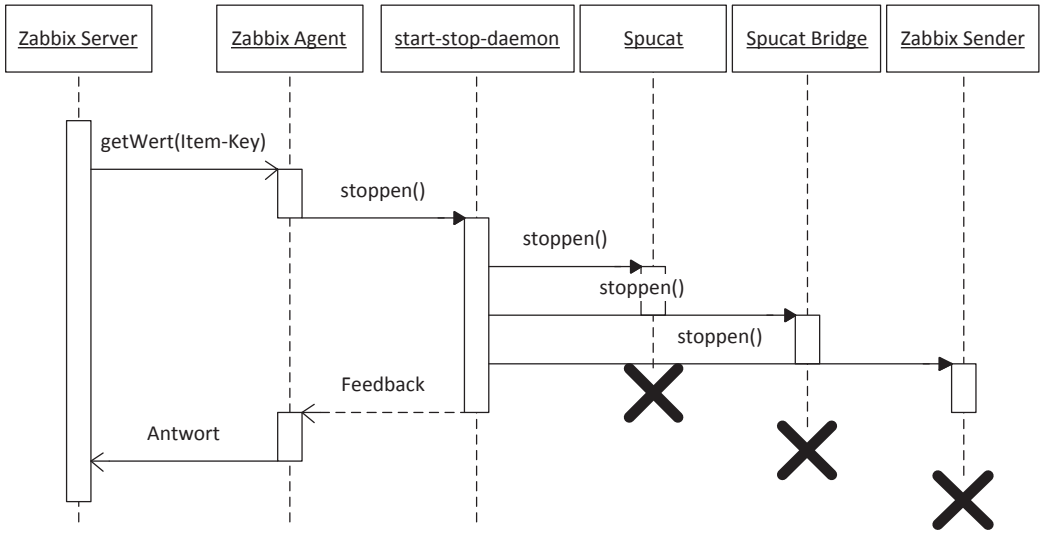

Abbildung 32 Sequenzdiagramm zum Stoppen der SHM Überwachung

4.3.4 Anpassungen der Weboberfläche

Die Weboberfläche von Zabbix visualisiert die eingegangen Daten innerhalb von verschiedenen Graphen. Die kleinste Skaleneinteilung zur Darstellung der Daten beträgt dabei 3600 Sekunden, d. h. ab einem gewählten Zeitpunkt werden alle zur Verfügung stehenden Daten der zurückliegenden Stunde innerhalb des Graphen visualisiert. Im Abschnitt 3.2.2 wurde verdeutlicht, dass diese Zeitspanne für eine effektive Analyse verringert werden muss, da die Daten einer Stunde nicht ohne eine weitere Verarbeitung, wie beispielsweise Mittelwertbildung für eingegangene Daten innerhalb der gleichen Minute, dargestellt werden können. Hierzu sind einige Anpassung und Erweiterungen der Weboberfläche vorzunehmen. Diese Änderungen beschränken sich auf drei Dateien, die im Folgenden näher besprochen werden sollen.

Frontend-Pfad/include/defines.inc.php

Diese Datei enthält grundlegende benannte Konstanten und definiert unter anderem die minimale Skaleneinteilung sowie die Standard-Einteilung der Graphen. Die minimale Skaleneinteilung wird dabei von einer Stunde auf 30 Sekunden verringert sowie die Standard-Einteilung auf eine Minute herabgesetzt:

```
28: define('ZBX_MIN_PERIOD', 3600);      → define('ZBX_MIN_PERIOD', 30);
30: define('ZBX_PERIOD_DEFAULT', 3600); → define('ZBX_PERIOD_DEFAULT', 60);
```

Frontend-Pfad/js/gtlc.js

Um die Zoom-Funktion sowie die Auswahl der darzustellenden Skaleneinteilung auch für den verkleinerten Zeitraum verwenden zu können, ist die Anpassung von mehreren Zeilen in dieser Datei notwendig. Als Ergebnis der Anpassung kann sowohl in einen Bereich von

30 Sekunden hineingezoomt werden, als auch kürzere Skaleneinteilungen zur Darstellung, wie in Abbildung 33 zu sehen, ausgewählt werden.

Abbildung 33 Auswahl der Skaleneinteilung in der Weboberfläche

Die Skriptdatei setzt den Standard-Wert der Skaleneinteilung und definiert ebenfalls eine Variable für die minimale Skaleneinteilung. Die Überprüfung der minimalen Skaleneinteilung findet nach Auswahl des Zoom-Bereiches statt, d. h. es wird überprüft, dass der Bereich in den hineingezoomt werden soll, nicht kleiner ist als die minimale Skaleneinteilung. Die entsprechenden Zeilen werden analog der benannten Konstanten aus der Datei „defines.inc.php„ angepasst:

```
60: [..] time.period = 3600;     → [..] time.period = 60;

409:     minperiod: 3600,        → minperiod: 30,
```

Zur Formatierung der Zeitangabe des dargestellten Zeitraumes oberhalb der Graphen sowie der auswählbaren Skaleneinteilungen, sind die entsprechenden Funktionen anzupassen. Hierzu wird die Berechnung der vollen Sekunden hinzugefügt, das Anhängen einer führenden Null bei einstelligen Sekunden ergänzt sowie die Formatierung der Ausgabezeiten für die Ausgabe von Sekunden und Minuten angepasst. Die Änderungen werden in den Funktionen „formatStampByDHM“ bzw. „formatStampbyDHM“ vorgenommen. Die Details der Anpassung werden in Anlage 7 beschrieben.

Um eine Auswahl der kürzeren Skaleneinteilungen zwischen 30 Sekunden und 15 Minuten zu ermöglichen, müssen diese in Form von Sekunden in der entsprechenden Variable hinzugefügt werden:

```
1422: var zooms = [3600, (2*3600), (3*3600), [..] ]; →
      var zooms = [30, 60, 300, 900, 3600, (2*3600), (3*3600), [..] ];
```

Frontend-Pfad/jsLoader.php

In dieser Datei sind für die Skriptdatei „gtlc.js“ die Kurzzeichen der Zeitbasen festgehalten, zu welchen das Kurzzeichen für Sekunden hinzugefügt wird:

```
93: 'S_SECOND_SHORT' => _x('s', 'second short'),
```

Bei der Analyse der Graphen zeigt sich eine Abweichung der Verläufe. In Abbildung 34 ist der Wertüberlauf zweier vorzeichenloser 32-bit Variablen zu sehen. Bei Überlauf der Variablen wird ein steiler und quasi senkrechter Abfall des Verlaufes erwartet, stattdessen weist der Verlauf im Abfall einen Zwischenwert auf.

Abbildung 34 Abweichung vom erwarteten Graphen-Verlauf

Der Grund hierfür liegt in der Datenbankabfrage, da innerhalb dieser die eingegangenen Werte nach einem berechneten Index auf Basis des Zeitstempels gruppiert werden und innerhalb dieser Gruppen lediglich die Mittelwerte sowie die Maxima und Minima berechnet und innerhalb der Graphen dargestellt werden. Somit wird über alle Werte, die den gleichen Index besitzen, der Mittelwert gebildet. Abhilfe schafft die zusätzliche Abfrage und Gruppierung der erfassten Nanosekunden sowie die Sortierung nach dem Zeitstempel und den Nanosekunden (Ergänzungen rot markiert):

```
SELECT itemid,
ROUND( 1776 * MOD( CAST( clock AS UNSIGNED ) + 24, 900 ) / ( 900 ) , 0 ) AS i,
COUNT( * ) AS count, AVG( value ) AS avg, MIN( value ) AS min,
MAX( value ) AS max, MAX( clock ) AS clock, ns
FROM history
WHERE itemid =23554
AND clock >=1345795176
clock 1345796076
GROUP BY itemid, ns,
ROUND( 1776 * MOD( CAST( clock AS UNSIGNED ) +24, 900 ) / ( 900 ) , 0 )
ORDER BY clock, ns ASC
```

Durch diese Ergänzungen bleibt die eigentliche Abfrage erhalten, lediglich die Gruppierung der Werte wird unterbunden und es werden alle gespeicherten Werte zurückgegeben. Die originale Datenbankabfrage findet dennoch bei Skaleneinteilungen, die größer sind als 5 Minuten, Anwendung. Zur Berechnung des Indizes „i“ ist die Breite des Graphen (im Beispiel 1776 Pixel) sowie die Skaleneinteilung (900s) notwendig. Die zur Verfügung stehende Breite des Graphen wird dabei effektiv genutzt, solange die Skaleneinteilung größer ist, als die Anzahl der Pixel, bei einer entsprechend hohen Anzahl an darstellbaren Daten. Einen Auszug aus der obigen Datenbankabfrage zeigt Tabelle 16. Bei einer Skaleneinteilung von 3600s werden alle zur Verfügung stehenden Indizes genutzt:

Tabelle 16 Indizes bei einer Skaleneinteilung von 3600s

i	count	avg	min	max	clock
0	43	205.465.004.651.163	20.352.000.000	20.661.760.000	1345798776
1	14	205.511.314.285.714	20.352.000.000	20.661.760.000	1345795179
2	29	205.442.648.275.862	20.352.000.000	20.661.760.000	1345795181
3	29	205.442.648.275.862	20.352.000.000	20.661.760.000	1345795183
4	29	205.442.648.275.862	20.352.000.000	20.661.760.000	1345795185
5	14	205.511.314.285.714	20.352.000.000	20.661.760.000	1345795187

Wird die Skaleneinteilung auf 900s gesenkt, bleiben Indizes, wie in Tabelle 17, ungenutzt obwohl ausreichend Daten zur Verfügung stehen, wie anhand der Anzahl zu sehen ist:

Tabelle 17 Indizes bei einer Skaleneinteilung von 900s

i	count	avg	min	max	clock
0	29	205.442.648.275.862	20.352.000.000	20.661.760.000	1345796076
2	15	205.378.560.000.000	20.352.000.000	20.661.760.000	1345795177
6	14	205.511.314.285.714	20.352.000.000	20.661.760.000	1345795179
8	15	205.378.560.000.000	20.352.000.000	20.661.760.000	1345795180

Um möglichst alle gespeicherten Werte innerhalb des Graphen darstellen zu können, werden diese nach der Abfrage mit einem neuen Index versehen, der die Daten gleichmäßig über die zur Verfügung stehende Breite des Graphen verteilt. Dabei kommt es vor, dass ein Bildpunkt nicht an der Position seines Zeitstempels dargestellt wird. Da die Zeitstempel erst im Zabbix Sender erfasst werden und somit einer Laufzeitverzögerung unterliegen sowie ein Auflösungsvermögen von einer Sekunde besitzen, ist dieser Umstand vernachlässigbar. Die Darstellung einer maximalen Anzahl von erfassten Daten hat dem gegenüber Priorität.

5 Bewertung

Der große Vorteil von „Zabbix“ gegenüber den anderen bewerteten Systemen liegt in der hohen Flexibilität der Software in Bezug auf Überwachungs- und Analysemöglichkeiten, die zu großen Teilen aus der umfangreichen Weboberfläche herrühren, wie im Abschnitt 4.1 zu lesen ist.

Im Auslieferungszustand unterstützt „Zabbix“ eine Vielzahl von Metriken zur Überwachung von Netzwerken und Serverressourcen, wie im Abschnitt 3.3.5 nachgelesen werden kann. Die Software erfüllt somit die meisten der beschriebenen Anforderungen aus Abschnitt 3.2.1 und erhält somit mit die meisten Punkte im Bereich der Überwachungskriterien. Darüber hinaus kann der Agent durch eigene Skripte und ausführbare Dateien erweitert werden, um neue Metriken einzubauen. Dabei ist lediglich zu beachten, dass die Erweiterungen ihre Daten innerhalb eines Zeitlimits auf der Standardausgabe wiedergeben. Bis auf das einheitliche Ausgabeformat der Daten ist somit eine lose Kopplung der Erweiterungen zur Überwachungssoftware möglich. Im Gegensatz zu den anderen bewerteten Überwachungssystemen ist so eine schnellere Erweiterung des Agenten möglich, da diese zum Teil eigene Schnittstellen für Erweiterungen definieren.

Ebenso wie die Vielzahl an unterstützen Metriken zur Überwachung bietet die Weboberfläche von „Zabbix“ eine große Vielfalt zur Auswertung und Analyse der Daten. „Zabbix“ ist dabei die einzige Überwachungssoftware, bei der die gespeicherten Daten in Tabellenform ausgegebenen werden können. Darüber hinaus können Trigger eingerichtet werden, mithilfe deren eingehende Daten auf die Einhaltung von festgelegten Grenzwerten überprüft werden können. Somit erfüllt „Zabbix“ einen Großteil der Anforderungen aus Abschnitt 3.2.2 und erhält im Vergleich der Überwachungssysteme die meisten innerhalb der dazugehörigen Prüfkriterien. Da die Darstellung der gespeicherten Daten für kleinere Skaleneinteilungen von weniger als einer Stunde nicht vorgesehen ist, musste die Weboberfläche entsprechend angepasst werden. Hierzu wurden verschiedene Dateien angepasst, um Skaleneinteilungen von bis zu 30 Sekunden zu ermöglichen. Ein Großteil der bewerteten Systeme unterstützt im Auslieferungszustand keine entsprechenden Zeitspannen. Bei System, die das Round-Robin Database Tool einsetzen, ist darüber hinaus die Anpassung der Daten notwendig. Bei der Auswertung der Graphen, zeigte sich eine Abweichung im Verlauf der Graphen vom erwarteten Wert. Um diese Abweichung zu korrigieren war hierzu eine Anpassung der Datenbankabfrage notwendig, da innerhalb dieser gespeicherte Werte gruppiert wurden und eine Mittelwertbildung stattgefunden hat. Details hierzu können dem Abschnitt 4.3.4 entnommen werden.

Das Einrichten der Metriken zur eigentlichen Überwachung erfolgt dabei ausschließlich über die Weboberfläche, ebenso wie die Konfiguration der Trigger und benutzerdefinierten

Graphen zur Auswertung der erfassten Daten. Auf diese Weise erfolgt die Einrichtung von einer zentralen Stelle aus und kann bei Bedarf auch von dieser geändert werden. Die Konfiguration der Überwachung kann so ohne einen Zugang zum Betriebssystem des überwachten Systems erfolgen. Ausnahme bildet hier die Installation und Konfiguration des Agenten bzw. Senders, wenn diese Anwendungen benötigt werden.

Darüber hinaus erfüllt „Zabbix“ weitere Anforderungen, die in Abschnitt 3.2.3 beschrieben wurden. Da sich der Überwachungsserver inklusive Datenbank auf einem separaten Gerät befinden kann, kann somit auch eine minimale Belastung der Verarbeitungssysteme gewährleistet werden. Abbildung 35 zeigt hierzu die Belastung der Überwachungskette für einen CPU-Kern auf einem Server. Ausgeführt wird jeweils ein Prozess mit einem Thread.

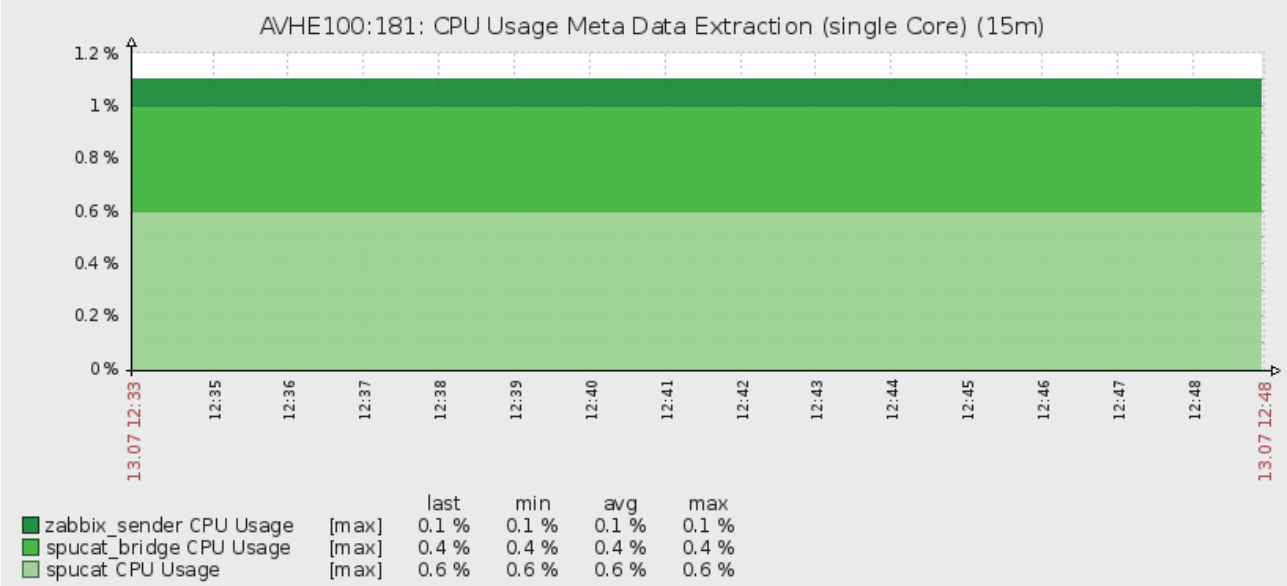

Abbildung 35 CPU-Belastung durch Überwachungskette für einen Kern

Ein CPU-Kern wird mit 1,1 % im Durchschnitt belastet. Dabei entfällt ein Großteil der Rechenzeit auf den Prozess von „Spucat“ sowie auf die Brückenanwendung „Spucat Bridge“. Die Rechenzeit der Brückenanwendung wird größtenteils für die Verarbeitung der Zeichenketten benötigt, um die enthaltenen Informationen verarbeiten zu können. Um Rechenzeit einzusparen, ist eine Fusion beider Prozesse denkbar. Dabei gibt die weiterentwickelte Spucat-Anwendung die Daten in einem Format aus, welches vom Sende-Prozess verarbeitet werden kann, um diese Daten an den Überwachungsserver zu senden. Somit können bis 0,4 % der Rechenzeit der Brückenanwendung eingespart werden. Dieser Umstand erhält zusätzliche Bedeutung, wenn eine parallele Überwachung mehrerer Shared Memories umgesetzt werden soll. Durch die Verkürzung der Überwachungskette würde sich ebenfalls deren Komplexität verringern und somit die Abhängigkeit der Brückenanwendung von der aktuellen Spucat-Implementierung entfallen.

Die Überwachung der gemeinsamen Speicher der Videoprozesskette bedarf einer häufige Änderung des Shared Memory Namens. Dieser wird, wie in Abschnitt 4.3.2 beschrieben wird, als Makro innerhalb der Hostkonfiguration gespeichert. Um eine aktuelle Überwachung auf einen neuen Shared Memory anzuwenden, muss hierzu zuerst die aktuell laufende Überwachung mittels Stop-Item beendet, anschließend der Name des Shared Memories angepasst und die Überwachung neugestartet werden. Das Problem hierbei liegt im Aufbau der Weboberfläche, da die Aktionen innerhalb der Menüs auf unterschiedlichen Seiten ausgeführt werden. Der Benutzer muss somit zwischen unterschiedlichen Menüs wechseln, was die Einrichtung der Überwachung verkompliziert. Des Weiteren muss der Shared Memory Name manuell eingegeben werden. Eine Auswahl per Drop-Down Menü der aktiven Shared Memories ist ohne umfangreiche Anpassungen der Weboberfläche nicht möglich. Diese Nachteile der Weboberfläche stammen aus dem eigentlichen Anwendungsgebiet der Überwachungssoftware: die Langzeit-Überwachung von großen Netzwerken und Serveranordnungen. Dabei erfolgt meist eine statische Konfiguration der Überwachung. Änderungen dieser erfolgen seltener und betreffen mehrere überwachte Komponenten. Abhilfe schafft an dieser Stelle die Entwicklung einer spezifischen Weboberfläche, die eine einfachere und schnellere Anpassung der Überwachung ermöglicht. Diese Weboberfläche kann dabei die API von „Zabbix“ nutzen, um die Überwachung zu starten bzw. wieder zu beenden sowie eingegangene Daten abzufragen. Denkbar ist ebenfalls, die manuelle Eingabe des Shared Memory Namens gegen ein Auswahlmenü zu tauschen. Hierzu muss zusätzlich die Konfiguration der Videoverarbeitungskette ausgelesen werden, um die aktiven Shared Memories zu ermitteln.

Abschließend kann festgehalten werden, dass mit „Zabbix“ als Monitoring-System eine Software gefunden wurde, mit deren Hilfe sowohl eine Überwachung der Server möglich ist, als auch eine spezifische Überwachung der Videoverarbeitungskette über die gemeinsamen Speicherbereiche der Anwendungen. Im direkten Vergleich mit anderen Systemen bietet „Zabbix“ die umfangreichsten Überwachungs- und Analysemöglichkeiten. Des Weiteren können zusätzliche Metriken durch eigene Erweiterungen eingepflegt werden. Der Umfang der notwendigen Anpassungen zum Einfügen kleinerer Skaleneinteilungen und Abfrage der gespeicherten Rohdaten hielt sich gering und beschränkt sich auf vier Dateien. Es zeigten sich zwar Schwächen bei der notwendigen häufigen Veränderung der Shared Memory Überwachung innerhalb der umfangreichen Weboberfläche, dies lässt sich zum Teil durch etwas Übung kompensieren, indem für die häufig benötigten Seiten ein eigener Tab im Browser geöffnet wird.

Zusammenfassung

Ziel dieser Diplomarbeit war es, ein geeignetes Open-Source-Monitoring-System für die Rohde & Schwarz GmbH & Co KG zu finden, welches unterstützend zur Analyse des Datenflusses zwischen einzelnen Prozessstufen einer Videoverarbeitungskette eingesetzt werden kann.

Zu diesem Zweck wurde die Videoverarbeitungskette hinsichtlich ihres Aufbaus sowie ihrer Struktur hin untersucht, um entsprechende Anforderungen zu erstellen und diese in die theoretischen Betrachtungen einzuordnen. Die so ermittelten Anforderungen flossen in einen Bewertungsmaßstab, der die gewählten System hinsichtlich ihrer Überwachungs-, und Analysefähigkeiten sowie deren technischen und nicht-technischen Merkmale beurteilt. Aus dieser Bewertung ging letztendlich das „Zabbix“ System als Favorit für den angestrebten Verwendungszweck hervor.

Die Videoverarbeitungskette nutzt gemeinsame Speicherbereiche, um Daten zwischen den einzelnen Prozessstufen auszutauschen. Um die Verbindung zu den gespeicherten Daten innerhalb dieser Speicherbereiche herzustellen, wird eine vorhandene Applikation eingesetzt, welche die Daten als aufbereitete Zeichenkette auf der Standardausgabe wiedergibt. Unter Zuhilfenahme einer eigens entwickelten Anwendung werden diese Ausgabedaten in ein Format konvertiert, welches vom Monitoring-System verarbeitet werden kann.

Die Auswertung der eingegangenen Daten erfolgt über die mitgelieferte Weboberfläche des „Zabbix“ Monitoring-Systems. Darüber hinaus wurde die Weboberfläche so eingerichtet, dass über diese eine Konfiguration der Überwachung sowie das Starten bzw. Beenden dieser ermöglicht wird. Um die eingegangenen Daten über die Weboberfläche adäquat analysieren zu können, war die Anpassung der Skaleneinteilungen der Graphen notwendig, da die Standardwerte für die Analyse von Audio- und Videodaten ungeeignet sind. Hierzu wurden zusätzliche kürzere Skaleneinteilungen hinzugefügt sowie Anpassungen der Datenabfrage vorgenommen, um eine Vorverarbeitung der gespeicherten Daten zu vermeiden und eine Darstellung der Rohdaten zu ermöglichen.

Abschließend kann festgestellt werden, dass mithilfe des „Zabbix“ Monitoring-Systems erfolgreich ein unterstützendes Werkzeug zur Analyse des Datenflusses der Videoverarbeitungskette integriert werden konnte. Während der Verwendung zeigten sich jedoch Nachteile innerhalb des Bedienungskonzeptes der Weboberfläche, da diese für eine statische Einrichtung der Überwachung entwickelt wurde und somit eine häufige Veränderung der Überwachungsparameter erschwert. Die Entwicklung einer speziellen Weboberfläche zur Steuerung der Überwachung von einem bzw. mehreren Speicherbereichen kann diesen Umstand beseitigen.

Quellenverzeichnis

[1] Bundesamt für Sicherheit in der Informationstechnik, „Hochverfügbarkeitskompendium,“ 31.05.2010. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Hochverfuegbarkeit/12_Monitoring_pdf.pdf?__blob=publicationFile. [Zugriff am 25.08.2012].

[2] D. C. Verma, Principles of Computer Systems and Network Management, Yorktown Heights: Springer, 2009.

[3] AT&T Inc., „History of the AT&T Network - History of Network Management,“ [Online]. Available: http://www.corp.att.com/history/nethistory/management.html. [Zugriff am 16.04.2012].

[4] A. Westerinen und W. Bumpus, „The Continuing Evolution of Distributed Systems Management,“ *IEICE Transactions on Information and Systems,* Nr. 11, S. 2256–2261, 2003.

[5] Case, Davin, Fedor und Schoffstall, „RFC 1028: A Simple Gateway Monitoring Protocol,“ Internet Engineering Task Force, 1987.

[6] Case, Davin, Fedor und Schoffstall, „RFC 1067: A Simple Network Management Protocol,“ Internet Engineering Task Force, 1988.

[7] McCloghrie und Rose, „RFC 1155: Structure and Identification of Management Information,“ Internet Engineering Task Force, 1990.

[8] McCloghrie und Rose, „RFC 1156: Management Information Base for Network Management of TCP/IP-based internets,“ Internet Engineering Task Force, 1990.

[9] Wikipedia, Die freie Enzyklopädie, „Distributed Management Task Force,“ Wikimedia Foundation Inc., 08 08 2010. [Online]. Available: http://de.wikipedia.org/wiki/Distributed_Management_Task_Force. [Zugriff am 18.04.2012].

[10] Bundesamt für Sicherheit in der Informationstechnik (BSI), „BSI: B 4.2 Netz- und Systemmanagement,“ 11 03 2010. [Online]. Available: https://www.bsi.bund.de/ContentBSI/grundschutz/kataloge/baust/b04/b04002.html. [Zugriff am 17.04.2012].

[11] E. Feess, „System,“ Springer Gabler | Springer Fachmedien Wiesbaden GmbH,

[Online]. Available: http://wirtschaftslexikon.gabler.de/Definition/system.html. [Zugriff am 23.04.2012].

[12] Host Europe GmbH, „Host Europe GmbH - Managed Hosting - Whitepapers - Service Level Agreements,“ 07.07.2009. [Online]. Available: http://www.hosteurope.de/download/ManagedHosting/Whitepapers/Whitepaper_SLA.pdf. [Zugriff am 24.04.2012].

[13] *Norm ISO/IEC 7498-4,* International Organization for Standardization, 1989.

[14] *Norm ITU-T X.700,* ITU Telecommunication Standardization Sector, 1992.

[15] R. W. W. Lahaye, „RFC 2613: Remote Network Monitoring MIB Extensions for Switched Networks,“ 1999.

[16] T. Schwenkler, Sicheres Netzwerkmanagement - Konzepte, Protokolle, Tools, Soest: Springer-Verlag Berlin Heidelberg, 2005.

[17] DATACOM Buchverlag GmbH, „MIB (management information base),“ 10.10.2011. [Online]. Available: http://www.itwissen.info/definition/lexikon/management-information-base-MIB.html. [Zugriff am 06.28.2012].

[18] McCloghrie und Rose, „RFC 1213: Management Information Base for Network Management of TCP/IP-based internets: MIB-II,“ Internet Engineering Task Force, 1991.

[19] Case, Davin, Fedor und Schoffstall, „RFC 1157: A Simple Network Management Protocol,“ Internet Engineering Task Force, 1990.

[20] Case, McCloghrie, Rose und Waldbusser, „RFC 1902: Structure of Management Information for Version 2 of the Simple Network Management Protocol,“ Internet Engineering Task Force, 1996.

[21] Case, McCloghrie, Rose und Waldbusser, „RFC 1906: Transport Mappings for Version 2 of the Simple Network Management Protocol,“ Internet Engineering Task Force, 1996.

[22] Case, McCloghrie, Rose und Waldbusser, „RFC 1907: Management Information Base for Version 2 of the Simple Network Management Protocol,“ Internet Engineering Task Force, 1996.

[23] Harrington, Presuhn und Wijnen, „RFC 3411: An Architecture for Describing Simple Network Management Protocol Management Frameworks,“ Internet Engineering

Task Force, 2002.

[24] Postel, „RFC 972: Internet Control Message Protocol,“ Internet Engineering Task Force, 1981.

[25] Lonvick, „RFC 3164: The BSD syslog Protocol,“ Internet Engineering Task Force, 2001.

[26] New und Rose, „RFC 3195: Reliable Delivery for syslog,“ Internet Engineering Task Force, 2001.

[27] W. Fischer, Digitale Fernseh- und Hörfunktechnik in Theorie und Praxis, München: Springer-Verlag Berlin Heidelberg, 2010.

[28] O. Vogel, I. Arnold, A. Chughtai, E. Ihler, T. Kehrer, U. Mehlig und U. Zdun, Software-Architektur, Heidelberg: Spektrum Akademischer Verlag, 2009, p. 229.

[29] J. Wolf, „Linux-UNIX-Programmierung – 9.3 Pipes,“ 2006. [Online]. Available: http://openbook.galileocomputing.de/linux_unix_programmierung/Kap09-002.htm#RxxKap09002040002D51F049100. [Zugriff am 2012.08.15].

[30] J. Garbis, P. Russell und D. Slama, Enterprise CORBA, Prentice Hall, 1999.

Verzeichnis der Anlagen

Anlagen

Anlage 1 AVHE100 System Template

Innerhalb dieses Templates sind verschiedene Items eingerichtet, die die Serverhardware überwachen. Die Werte der Items aus Tabelle 18 werden vom Zabbix Agenten direkt auf dem Server gemessen.

Tabelle 18 Items des AVHE100 System Templates

Item-Name	Schlüssel	Erläuterung
AVHE CPU Usage all, user, avg1	system.cpu.util [all,user,avg1]	Erfasst die prozentuale Auslastung aller CPU-Ressourcen (all) als Durchschnittswert der letzten Minute (avg1), die durch Benutzerprozesse (user) belegt werden
AVHE Network Incomming Traffic eth1	net.if.in [eth1,bytes]	Erfasst das eingehende Datenaufkommen auf dem Netzwerkinterface 1 (eth1), speichert die Veränderung zum vorherigen Wert, um Übertragungsrate zu erhalten.
AVHE Network Incomming Traffic eth0	net.if.in [eth0,bytes]	Erfasst das eingehende Datenaufkommen auf dem Netzwerkinterface 0 (eth0), speichert die Veränderung zum vorherigen Wert, um Übertragungsrate zu erhalten
AVHE Network Outgoing Traffic eth1	net.if.out [eth1,bytes]	Erfasst das ausgehende Datenaufkommen auf dem Netzwerkinterface 1 (eth1), speichert die Veränderung zum vorherigen Wert um Übertragungsrate zu erhalten
AVHE Network Outgoing Traffic eth0	net.if.out [eth0,bytes]	Erfasst das ausgehende Datenaufkommen auf dem Netzwerkinterface 0 (eth0), speichert die Veränderung zum vorherigen Wert, um Übertragungsrate zu erhalten
AVHE Total Memory	vm.memory.size [total]	Erfasst die Kapazität des gesamten physikalischen Speichers
AVHE Used Memory	vm.memory.size [used]	Erfasst die Höhe des belegten Speichers

Zur Analyse der CPU-Auslastung ist ein Trigger eingerichtet („Trigger CPU Usage“), der seinen Status ändert, sobald der Mittelwert über die CPU-Auslastung aus den letzten 120 Sekunden kleiner ist, als der Wert, der über das Host-Makro {$CPU_USAGE} festgelegt ist. Wenn dies der Fall ist, kann es ein Indiz dafür sein, dass die Prozesskette

unterbrochen ist und die Prozesse mit hohem Ressourcenverbrauch, wie beispielsweise der Video-Encoder, nicht arbeiten.

Zur Visualisierung der erfassten Daten sind die Graphen aus Tabelle 19 eingerichtet.

Tabelle 19 Graphen des AVHE100 System Templates

Graphen-Name	Erläuterung
Server CPU Usage All	Visualisiert die Auslastung der CPU-Ressourcen, die mit dem Item AVHE CPU Usage all, user, avg1 erfasst wurden
Server Memory	Stellt die Höhe des belegten Speichers dar, welche mit dem Item AVHE Used Memory erfasst wird. Als maximaler Wert wird der Wert es Items AVHE Total Memory verwendet
Server Network Traffic Eth0	Visualisiert den eingehenden sowie den ausgehenden Datenverkehr auf dem Netzwerkinterface 0, welcher über das Item AVHE Network Incomming Traffic eth0 sowie das Item AVHE Network Outgoing Traffic eth0 erfasst wird
Server Network Traffic Eth1	Visualisiert den eingehenden sowie den ausgehenden Datenverkehr auf dem Netzwerkinterface 1, welcher über das Item AVHE Network Incomming Traffic eth1 sowie das Item AVHE Network Outgoing Traffic eth1 erfasst wird

Anlage 2 SHM Common Template

Mithilfe des „SHM Common Templates“ werden die generischen Werte eines Shared Memories erfasst. Dazu gehören die Header- und Slot-Informationen, die häufigen Meta-Daten sowie die aus diesen Daten errechneten Werte. Der Zabbix Server erhält die Werte dieses Templates vom Zabbix Sender. Daher sind die festlegten Schlüssel aus Tabelle 20 von besonderer Bedeutung, denn der Sender ist hier für die Zuordnung von Wert zu Schlüssel zuständig.

Tabelle 20 Items des SHM Common Templates

Item-Name	Schlüssel	Erläuterung
SHM Data Offset	shm.data.offset	Der absolute Versatz des ersten Bytes der Nutzdaten zum Beginn des Shared Memories
SHM DTS	shm.dts	Decoding Timestamp
SHM DTS Delta	shm.dts.delta	Differenz des aktuellen DTS zum Vorhergehenden
SHM Duration	shm.duration	Dauer der Daten innerhalb des Slots

SHM Processing Delay	shm.common.delay	Geschätzte Verzögerung zwischen Datenempfang und -senden in Millisekunden [ms]
SHM PTS-DTS Delta	shm.pts.dts.delta	Differenz des letzten Presentation Timestamps zum aktuellen Decoding Timestamp
SHM PTS Spacing	shm.pts.spacing	Differenz des aktuellen PTS zum Vorhergehenden
SHM Reference TS	shm.ref.ts	Reference Timestamp
SHM Reference TS Delta	shm.ref.ts.delta	Differenz des aktuellen RTS zum Vorhergehenden
SHM TS	shm.time.stamp	Timestamp (auch Presentation Timestamp)
SHM TS Delta	shm.time.stamp.delta	Normierte Differenz des aktuellen PTS zum Vorhergehenden
SHM Type	shm.type	Typ der enthaltenen Daten
SHM Valid Time	shm.valid.time	Gültiger Zeitstempel des Shared Memories
SHM VaVi Delta	shm.vavi.delta	Differenz des Valid Timestamp zum Virtual Timestamp
SHM Virtual Time	shm.virtual.time	Virtueller Zeitstempel des Shared Memories

Zur Überwachung der eingehenden Daten sind drei Trigger eingerichtet. Die Trigger „SHM PTS Spacing Neg“ bzw. „SHM PTS Spacing Pos“ ändern ihren Status, sobald der Betrag von „SHM PTS Spacing“ 700 ms überschreitet. In solch einem Fall liegt ein Presentation Timestamp Fehler vor. Der Trigger „SHM VaVi Delta Trigger” überwacht die Werte des Items „SHM VaVi Delta“. Dieser darf nicht kleiner als 0 werden, andernfalls ist der Inhalt des Shared Memories ungültig. Aufgrund der einfachen Gestaltung der Daten wird in dem entsprechenden „Common Screen“ auf die einfachen Graphen zur Visualisierung zurückgegriffen. Zur Gegenüberstellung des Virtual Timestamps zum Valid Timestamp ist ein Graph „SHM VaVi“ eingerichtet, der den Kurvenverlauf beider Zeitstempel gegenüberstellt (siehe Abbildung 36).

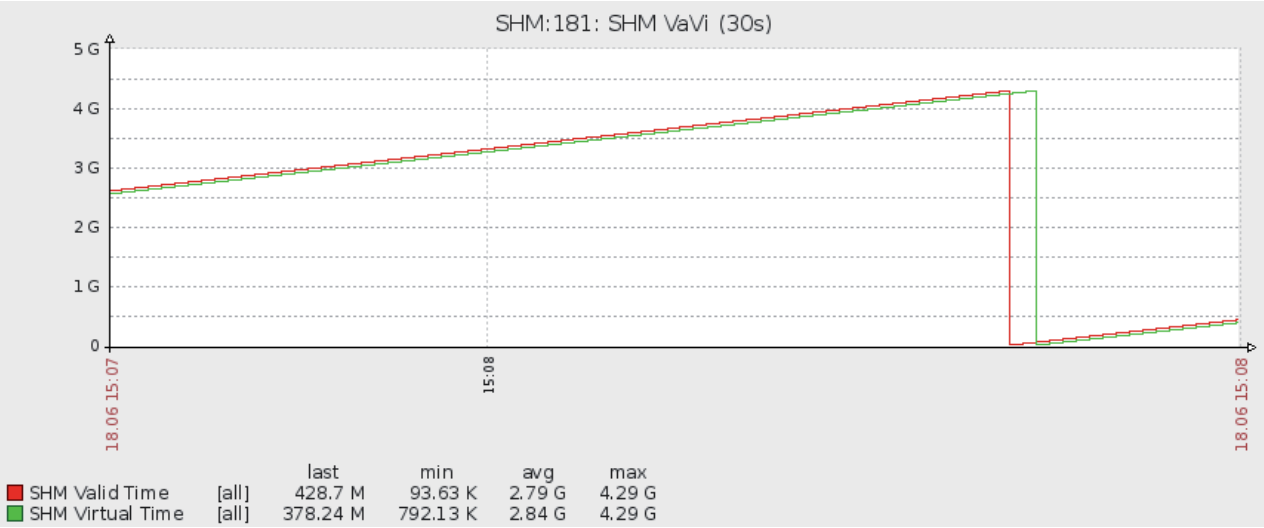

Abbildung 36 Verlauf des gültigen und virtuellen Zeitstempels

Im Verlauf der Anwendung kann es, wie in Abbildung 36, scheinbar dazu kommen, dass die Kurve des Valid Timestamps unterhalb der Kurve des Virtual Timestamps liegt und somit die Differenz kleiner als 0 ist. An dieser Stelle kommt es zu einem Überlauf des Zahlenbereiches des 32bit vorzeichenlosen Zählers. Die Differenz der beiden Zähler wird in einer 32bit vorzeichenbehafteten Variable gespeichert. Ist dabei die Differenz nicht größer als $2^{31} - 1$, bleibt sie im positiven Wertebereich.

Anlage 3 SHM Compressed Video Template

Mithilfe dieses Templates werden die spezifischen Werte eines Shared Memories, welches komprimiertes Videomaterial enthält, erfasst. Wie auch beim „SHM Common Template“ werden die Daten vom Zabbix Sender an den Server gesendet.

Tabelle 21 Items des SHM Compressed Video Templates

Item-Name	Schlüssel	Erläuterung
ES Bitrate	shm.com.vid.bitrate	Die Bitrate des Elementarstromes in Bit/s

Anlage 4 SHM Container Template

Das Item des „SHM Container Template“ erhält die Messwerte eines Shared Memories, wenn es Container-Daten enthält.

Tabelle 22 Items des SHM Container Templates

Item-Name	Schlüssel	Erläuterung
Bitrate	shm.container.bitrate	Die Bitrate des Stromes in Bit/s

Anlage 5 SHM Uncompressed Audio Template

Mithilfe des „SHM Uncompressed Audio Template” werden die Messwerte eines Shared Memories, welches unkomprimierte Audiodaten enthält, erfasst. Der Zabbix Server erhält die Daten vom Zabbix Sender und speichert diese unter dem mitgelieferten Schlüssel ab (siehe Tabelle 23).

Tabelle 23 Items des SHM Uncompressed Audio Templates

Item-Name	Schlüssel	Erläuterung
Channel	shm.uncom.aud.channel	Anzahl der Audiokanäle (1 für Mono, 2 für Stereo)
Sample Rate	shm.uncom.aud.rate	Abtastfrequenz in Hz

Anlage 6 SHM Uncompressed Video Template

Mithilfe des „SHM Uncompressed Video Template” werden die Messwerte eines Shared Memories, welches unkomprimierte Videodaten enthält, erfasst.

Tabelle 24 Items des SHM Uncompressed Video Templates

Item-Name	Schlüssel	Erläuterung
Aspect Denominator	shm.uncom.vid.aspect.den	Nenner des Sample-Seitenverhältnisses
Aspect Numerator	shm.uncom.vid.aspect.num	Zähler des Sample-Seitenverhältnisses
Display Aspect Ratio	shm.uncom.vid.dar	Berechnetes Anzeigeseitenverhältnis
Video Height	shm.uncom.vid.height	Höhe des Videos in Pixel
Video Width	shm.uncom.vid.width	Breite des Videos in Pixel

Anlage 7 Änderungen der Formatierungsfunktionen für Zeitangaben

Funktion „formatStampByDHM”:

Nach Zeile 1371 wird die Berechnung der vollen Sekunden hinzugefügt:

```
1371: var seconds = parseInt(timestamp - years*365*86400 - months*30*86400 –
       weeks*7*86400 - days*86400 - hours*3600 - minutes*60,10);
```

Der optionale Parameter mit dem Wert 10 der Funktion „parseInt“ beschreibt die Basis der zu interpretierenden Zahl. Dieser Parameter wird zusätzlich zur Typsicherung allen vorhergehenden Funktionen angehangen.

Nach Zeile 1378 wird das Anhängen einer führenden Null bei einstelligen Sekundenangaben ergänzt:

```
1378: if(seconds.toString().length == 1) seconds = '0'+seconds;
```

Der Code in Zeile 1389:

```
1389: str+=hours+locale['S_HOUR_SHORT']+'
           +minutes+locale['S_MINUTE_SHORT']+' ';
```

wird vollständig durch die folgenden Zeilen zur Formatierung der Ausgabezeiten ersetzt:

```
1389: str+=(hours == 0)?(''):(hours+locale['S_HOUR_SHORT']+' ');
1390: str+=(minutes == 0)?(''):(minutes+locale['S_MINUTE_SHORT']+' ');
1391: str+=seconds+locale['S_SECOND_SHORT']+' ';
```

Dadurch wird bewirkt, dass keine Angaben zu Stunden und Minuten ausgegeben werden, wenn diese gleich null sind.

Funktion „FormatStampbyDHM“:

Nach Zeile 2149 wird die Berechnung der vollen Sekunden hinzugefügt:

```
2149: var seconds = parseInt(timestamp - days*86400 - hours*3600 –
           minutes*60,10);
```

Der Code in Zeile 2152:

```
2152: str+=hours+'h '+minutes+'m ';
```

wird vollständig durch die folgenden Zeilen zur Formatierung der Ausgabezeiten ersetzt:

```
1389: str = (hours==0)?(''):(hours+'h ');
1390: str+= (minutes==0)?(''):(minutes+'h ');
1391: str+= seconds+'s ';
```

Dadurch wird bewirkt, dass keine Angaben zu Stunden und Minuten ausgegeben werden, wenn diese gleich null sind.