h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

Hochschule Darmstadt
- Fachbereich Informatik -

Der Titel der Arbeit

Abschlussarbeit zur Erlangung des akademischen Grades
Bachelor of Science (B.Sc.)

vorgelegt von

Vorname Nachname
12345

Referent:	Prof. Dr. Max Mustermann
Korreferent:	Dr. Max Muster

Abstract

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

> Dies ist ein Zitat.

verstand, scheinen nun doch vorueber zu Dies ist der Text sein.
siehe: http://janeden.net/die-praeambel

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listingverzeichnis

Kapitel 1

Einführung

Hallo sehr geschätzter Leser Willkommen bei meiner Bachelor Thesis in der ich den die Differenz zwischen Virtuellen Maschinen und realen Maschinen in einem Netzwerk vergleichen möchte. Dazu habe ich selber ein Netzwerktestframework entwickelt und von Netzwerktestframeworks demonstrieren, die in Netzwerken die aus realen Maschinen bestehen und aus virtuellen Maschinen verwendet werden. Wieso hat das eine relevanz? Das werde ich euch erklären wieso. Wir leben in einer immer weiter hochvernetzten Welt sind und die Industrie 4.0 wird immer mehr bestandteil unserer Welt, Kühlschränke werden mit dem Internet verbunden. Um eine hohe netzstabilität gewährleisten müssen wir.

Kapitel 2

Das Framework

2.1 Verwendete Hardware

Die in Abschnitt 4.1 beschriebenen Versuche wurden mit folgender Hardware gemacht.

- Zwei Switches
- Vier Raspberry Pis der ersten Generation
- Ein Raspberry Pi der zweiten Generation
- Mehrere Ethernet Kabel

Diese Geräte wurden in einem eigenständigend Netzwerk zusammengeschaltet. So sind an jedem Switch zwei Pis der ersten Generation angeschlossen während an einem der Switches der Pi der zweiten Generation angeschlossen ist siehe abbildung hier. Welche Software auf den Pis verwendet wurde, wird in Abschnitt 2.2 erklärt.

2.2 Aufbau der Software

Das Testframework besteht im Grunde aus drei verschieden Teilen.

1. Zabbix
2. Eigentwicklung auf dem Server
3. Eigentwicklung auf dem Agent

Die Eigentwicklungen sind alle in Bash programmiert, während Zabbix eine bereits fertige Open Source Lösung ist. In den folgenden Abschnitten werde ich einen Einblick in diese Teile geben.

2.2.1 Zabbix

Zabbix ist ein Open Source Netzwerk Monitoring System. Die erste Version wurde von Alexei Vladishev entwickelt, welches inzwischen von der Firma Zabbix SIA weiterentwickelt wird. Ein weiterer bekannter Vertreter der Netzwerk Monitor Systeme ist Nagios, welches wie Zabbix unter der GPL Lizenz vertrieben wird. Beide Systeme basieren auf einer Client-Server Architektur. Im weiteren wird jedoch nur Zabbix betrachtet. Zabbix besteht aus zwei Komponenten.

Zabbix Server Der Server hat eine auf PHP basierende Weboberfläche über die es für den Benutzer möglich ist die Agents zu konfigurieren. So können manuell die Templates erstellt werden die den Zabbix Agents mitteilen welche Informationen, dem Server zu übermitteln sind. Über die API Schnittstelle ist es möglich Programme für den Server zu schreiben und diese auszuführen. Der Zabbix Server selber ist auch ein Agent welcher sich selber überwacht. Dieser ist jedoch nicht beteiligt im Allgemeinen Prozess des Frameworks.

Zabbix Agent Die Clients die im Netzwerk überwacht werden sollen sind sogenannte Agents die an den Server die Informationen weiterleiten, die vom Server gefordert werden.

2.2.2 Eigenentwicklung

Die Selbstentwickelte Software wird in zwei Kategorien unterteilt so läuft ein Teil der Software auf den Agents diese haben den Zweck Netzwerk Traffic zu erzeugen und somit Daten die vom Server gesammelt werden kann.

1. Zabbix Server

   Update Script Dieses Script wird von der Software Entwicklungsmaschine ausgeführt. Es aktualisiert den Code auf den Endgeräten die die Programme Hintergrundrauschen Pinger und Startrauschen ausführen. Dabei wird mittels Secure Copy der Quellcode auf das Endgerät gespielt

   Get logs Die Skripte Pinger und Hintergrundrauschen erstellen, jeweils auf den Endgeräten logfiles. Da es jedoch ein sehr hoher Verwaltungsaufwand wäre auf den Endgeräten die Logfiles weiterzuverwerten werden mit dem Skript GetLogs die dezentralisiert gelagerten Logfiles auf dem Rechner der dieses Skript gestartet gesammelt. Somit hat man die von den Endgeräten gesammelten Daten auf einem Rechner und kann dessen weiterverarbeitung betreiben.

   Calculate Average Berechnet den Durschnitt der dauer die ein Paket brauch um erfolgreich versandt zu werden.

2. Zabbix Agent

   Hintergrundrauschen Diese Eigentwicklung stellt mit Zabbix die Kernkomponente des Frameworks dar. Wie der Rest der selbstentwickelten Software wurde sie komplett in

Bash programmiert da das Framework ausschließlich in einer Linux Umgebung entwickelt, getestet und verwendet wurde. Hintergrundrauschen schickt über die Linux Standardbefehle Secure Copy, eine Abwandlung der SSH, Pakete von einem Endegerät zum anderen. So wird eine Last auf dem Netzwerk erzeugt die mithilfe von Zabbix gemessen werden kann. Ausserdem speichert dieses Programm die Dauer bis ein Paket erfolgreich bei seinem, zufällig ausgewählten Empfänger angekommen ist und dessen größe.

Startrauschen wird automatisch zum Start eines der Endgeräte ausgeführt. Es dient als eine Zeitschaltuhr und ermöglicht ein Zeitversetztes starten des Scripts Hintergrundrauschen, wodurch man den sequentiellen Anstieg an Last im Ethnernet beobachten kann.

Synchronize Raspberry Pis besitzen keine eigene Batterie wie es handelsübliche Rechner haben deshalb ist nach jedem Neustart die Zeit der Pis unzuverlässig. Dieses Programm aktualisiert die Zeiten der Agents und synchronisiert die Zeit der Raspberry Pis mit der vom Zabbix Server.

Pinger wird zusammen mit Hintergrundrauschen ausgeführt und ist um den Linux eigenen Ping Befehl aufgebaut. Mittels Pinger wird die Latenz unter den einzelnen Endgeräten gemeßen.

2.3 Einsatz im Netzwerk

Mit der im vorherigen Abschnitt vorgestellten Software ist das Testframework aufgebaut. In der Abb. 2.1 wird dargestellt wie die einzelnen Komponenten zusammenspielen. Hier sieht man das an einem Switch zwei Raspberry Pis angeschlossen sind und an einem anderen Switch drei Raspberry Pis sind. Einer von diesen Pis ist der Zabbix Server der die Aktiven Hosts im folgenden Agents/Zabbix Agents im Netzwerk überwacht.

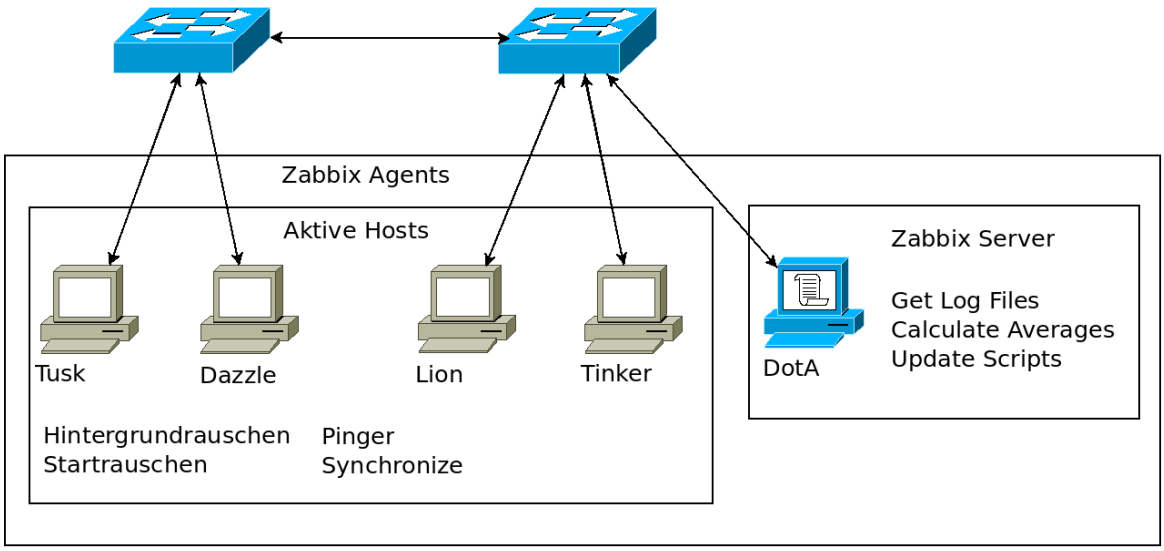

Abbildung 2.1: Aufbau vom Netzwerk

Kapitel 3

Testaufbau

3.1 Pis Real

3.2 Pis Virtuell

Kapitel 4

Versuche

Die beste Art ein Netzwerktestframework zu Testen ist in dem man das Framework benutzt dazu habe ich am Anfang der Arbeiten selber mehrere Tests definiert, jeder dieser Tests ist eine in einem Netzwerk mögliche Fehlerquelle die die Leistung des Netzwerkes beeinträchtigen kann. Um Fehler identifizieren zu können müssen jedoch auch die Normalwerte bestimmt werden. Dazu muss erstmal ein Störungsfreies Netzwerk aufgebaut werden und dieses muss beobachtet werden. Die in diesem Netz gesammelten Werte stellen die Basis für unsere späteren Ergebnisse dar. Der zweite Schritt ist es das Netz mit Fehlern aufzubauen und zu beobachten. Dadurch haben wir die Werte eines Fehlerbehafteten Netzwerkes und können diese nun mit dem fehlerfreien Netzwerk vergleichen. Dadurch sollte es uns nun möglich sein automatisierte Aussagen über die Art des Fehlers in einem Netzwerk zu treffen.

4.1 Versuchsbeschreibung

Normalbetrieb Dieser Test erzeugt die Testwerte mit denen die Fehlerbehafteten Tests verglichen werden.

Ethernetkabel ohne Isolierung Ethernetkabel werden oft sehr strapaziert dies kann dazu fuhren das sich die Isolation löst und somit Strahlung einwirkt. Dies kann die Leistung im Netzwerk beeinträchtigen.

Falsch gedrehtes Twisted Pair Kabel Twisted Pair Kabel gehören zu den gängigsten Kabeln des Ethernet Standards, welches durch eine Verdrillung der einzelnen Kabel einen erhöhten Einstrahlungsschutz bilden. [Tan03]

Loop Ein Loop ist wenn ein Switch mit sich selbst verbunden ist, der Switch sendet dann ständig Datagramme an sich selbst und blockiert so jeden Traffic auf der Leitung.

Nicht angeschlossenes Kabel Wie verhält sich das Netzwerk wenn ein Kabel von einem Endgerät im laufenden Betrieb entkoppelt wird.

Forkbomb Die Forkbomb ist ein Programm das von sich rekursiv Kopien erstellt so das der Computer alle sein Resourcen verbraucht nur noch diesen Code auszufuhren.

:(){ :|:&};: [Wik15]

Festplatte läuft voll In einem Netzwerk werden ständig Daten versand, was also passiert wenn die Festplatte von einem Rechner volläuft und der PC somit Arbeitsunfähig wird.

IP Adresse doppelt belegt Firmennetze benutzen statische IPs damit Webserver immer unter dem selben Namen erreichbar bleiben, das jedoch zu konfigurieren kann zu Fehlern Menschlichen Fehlern fuhren, was passiert in einem Netz wenn zwei Rechner sich eine IP teilen.

Kollisionsdomänen Kollisionsdomänen sind bereiche in einem Netzwerk wo sich mehrere Rechner eine eine Leitung teilen, dies ist geschieht in der Physikalischen Ebene eines Netwerks. Die Frage ist kann man eine Kollisionsdomäne erkennen.

4.2 Raspberry Pi Versuche

4.2.1 Gemachte Tests

4.2.2 Ergebnisse

4.3 Virtual Machine Versuche

4.3.1 Gemachte Tests

4.3.2 Ergebnisse

Kapitel 5

Vergleich VM/HW

5.1 Versuchsergebnisse

5.2 Kosten nutzen Faktor

Kapitel 6

Fazit

Literatur

[Tan03] Andrew S. Tanenbaum. *Computernetzwerke*. PEARSON Studium, 2003.

[Wik15] Wikipedia. *Forkbomb — Wikipedia, Die freie Enzyklopadie*. [Online; Stand 11. Mai 2015]. 2015. URL: `https://de.wikipedia.org/wiki/Forkbomb`.