Frankfurt University of Applied Sciences
Fachbereich 2: Informatik und Ingenieurwissenschaften

Bachelorthesis

Entwicklung, Aufbau und Evaluation eines Verbunds aus Einplatinencomputern zur Simulation eines Internet-Knotens

zur Erlangung des akademischen Grades
Bachelor of Science

Studiengang
– Informatik (B.Sc.) –

Autor:	Maximilian Willner
Matrikelnummer:	1052866
Hauptreferent:	Prof. Dr. Christian Baun
Korreferent:	Prof. Dr. Eicke Godehardt
Abgabedatum:	07.01.2022

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Schriften entnommen sind, sind als solche kenntlich gemacht. Die Arbeit hat in gleicher Form noch keiner anderen Prüfbehörde vorgelegen.

Ort, Datum	Unterschrift Maximilian Willner

Zusammenfassung

Diese Bachelorthesis beschäftigt sich mit der Implementierung eines Internet-Knotens auf minimaler Hardware, geeignet für Demonstrationszwecke. Unter Verwendung von Routing-Software und einem Ethernet Switch wird ein Verbund aus Einplatinencomputern implementiert und die Funktionen des Border Gateway Protokolls dargestellt. Es werden Grundlagen des Internet-Routings und Autonomer Systeme sowie Aspekte eines Internet-Knotens beleuchtet. Zudem werden die eingesetzte Hard- und Software genauer untersucht und ein Projekt zur Simulation Autonomer Systeme verglichen. Designentscheidungen und Anforderungen an den Testaufbau werden dargestellt und die Implementierung des Verbunds vorgestellt. Im Verlauf der Arbeit wird gezeigt, wie die Geräte im Verbund miteinander kommunizieren und es wird unter Verwendung von Messungen die Funktionalität des Verbunds verifiziert. Schließlich wird der Versuchsaufbau und das gewählte Design bewertet und ein Fazit gezogen. Am Ende wird ein Ausblick auf zukünftige Erweiterungen gegeben.

Abstract

This bachelor thesis deals with the implementation of an Internet Exchange Point on minimal hardware, suitable for demonstration purposes. Using routing software and an Ethernet switch, a network of single-board computers is implemented and the functions of the Border Gateway Protocol are shown. Basics of Internet routing and autonomous systems as well as aspects of an Internet Exchange Point are highlighted. In addition, the hardware and software used are examined more closely and a project to simulate autonomous systems is compared. Design decisions and requirements for the test setup are illustrated and the implementation of the network is presented. In the course of the work it is shown how the devices in the network communicate with one another and the functionality of the network is verified using measurements. Finally, the setup and the selected design are evaluated and a conclusion is drawn. At the end, an outlook on future extensions is given.

Inhaltsverzeichnis

Abkürzungsverzeichnis

AS	Autonomes System
BGP	Border Gateway Protokoll
CDN	Content Delivery Network
CIDR	Classless Inter Domain Routing
DHCP	Dynamic Host Configuration Protocol
Euro-IX	European Internet Exchange Association
HDMI	High Definition Multimedia Interface
IANA	Internet Assigned Numbers Authority
ICMP	Internet Control Message Protocol
IP	Internet Protokoll
IS-IS	Intermediate System to Intermediate System
ISP	Internet Service Provider
IX-F	Internet Exchange Federation
IXP	Internet Exchange Point
LAN	Local Area Network
OSPF	Open Shortest Path First
RFC	Request For Comment
RIPE NCC	Réseaux IP Européens Network Coordination Centre
RIR	Regional Internet Registry
SD	Secure Digital
SSH	Secure Shell
TCP	Transmission Control Protokoll
UDP	User Datagram Protocol
USB	Universal Serial Bus
WLAN	Wireless Local Area Network

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Das Internet ist ein Netzwerk aus Netzwerken [7]. Es besteht aus vielen miteinander verbundenen Netzen. Diese Netze, auch Autonome Systeme (AS) genannt, kommunizieren miteinander mittels des Border Gateway Protokolls (BGP) und tauschen in BGP-Sitzungen Erreichbarkeitsinformationen aus, also Pfade zu Netzen, die über das austauschende Netz erreichbar sind (inklusive ihr eigenes). Dabei schließen sich bis zu tausende AS zwischen allen existierenden Netzwerkklassen, von Internetdienstanbietern, im Englischen Internet Service Provider (ISP), über Content Delivery Networks (CDN) bis hin zu Akademie- und Unternehmens-Netzwerken mittels Peering an Internet-Knoten zusammen [23] [1].
Laut PeeringDB, einer Datenbank über Peering-Informationen, existieren über 920 Internet-Knoten (Austauschpunkte) weltweit an denen sich über 42.000 AS zusammenschalten[1]. In Relation zur Menge von über 108.000 AS weltweit[2] stellt die Anzahl an zusammengeschalteten AS an Internet-Knoten einen großen Anteil dar.
Aufgrund dieser Relevanz erscheint es bedeutsam, die Funktionalität von Internet-Knoten in der Forschung, Lehrveranstaltungen oder Verkaufsgesprächen darstellen zu können. Um einen Einblick in den Betrieb des Internet-Knotens aus Betreiber- und Kundensicht[3] geben zu können, eignet sich ein Aufbau eines Testszenario im Rahmen eines praktischen Kurses. In dieser Bachelorthesis soll sich daher mit der Frage beschäftigt werden, ob es möglich ist, einen Internet-Knoten in einem Versuchsaufbau zu realisieren, die Funktionen eines Internet-Knotens und die Kommunikation zwischen den Netzwerken darzustellen. Dazu soll untersucht werden, wie man mit einem Verbund aus einzelnen Geräten die Funktionalität eines Internet-Knotens darstellen kann. Hierbei können Einplatinencomputer eine kostengünstige Plattform für praktische Übungen oder Lehrveranstaltungen darstellen.
Es wird auf allen verbundenen Einplatinencomputern ein Betriebssystemabbild sowie

[1] https://www.peeringdb.com/
[2] https://www-public.imtbs-tsp.eu
[3] Genderhinweis: Aus Gründen der Lesbarkeit wird in dieser Bachelorthesis auf Gendering verzichtet. Bei allen geschlechtsspezifischen Begriffen sind fortlaufend immer alle Geschlechter gemeint.

Netzwerk-Software aufgespielt. Zudem wird eine Netzwerkkonfiguration gewählt, die für die Erfüllung der oben genannten Problemstellung geeignet ist. Es wird veranschaulicht, wie zwischen den verbundenen Netzwerken die Funktion eines Internet-Knotens abgebildet werden kann.
Um eine Grundlage für die Thematik zu schaffen, wird in Kapitel zwei zunächst auf die verwendeten Begrifflichkeiten und Definitionen eingegangen. Kapitel drei beinhaltet eine Vorstellung über die in dieser Bachelorthesis verwendeten aktuellen Technik und der verwendeten Kommandozeilenwerkzeuge. Es wird zudem ein Projekt vorgestellt, welches einen ähnlichen Ansatz für Lehrveranstaltungen wählt. In Kapitel vier folgt die Beschreibung des Designs des Versuchsaufbaus. Es werden Anforderungen an den Versuch sowie benötigte Hardware und Software aufgelistet. Die gewählte Netzwerkarchitektur wird hierbei beleuchtet und der Versuchsablauf beschrieben. In Kapitel fünf wird auf die Implementierung des Verbunds aus Einplatinencomputern und auf die Konfiguration des Netzwerks und der Software eingegangen. Es werden Messungen durchgeführt und Ergebnisse präsentiert. Im vorletzten Kapitel werden schließlich der Versuchsaufbau und die gemessenen Daten bewertet. Am Ende wird ein Fazit gezogen und ein Ausblick gegeben.

2. Grundlagen

Um die Bedeutung von Internet-Knoten einzuordnen, folgt in diesem Kapitel eine einleitende Übersicht über das Internet Routing beziehungsweise wie die verschiedenen Netze im Internet miteinander kommunizieren. Dies dient dazu, das weitere Verständnis für die Arbeit zu gewährleisten und die Grundlage zu schaffen, auf der sich mit der Forschungsfrage fundiert auseinandergesetzt werden kann. Zu Beginn werden Definitionen vorgestellt und Begrifflichkeiten erläutert.

2.1. Internet Protokoll

Um eine Grundlage für diese Arbeit zu schaffen, wird zunächst kurz das Internet Protocol, zu Deutsch Internet Protokoll (IP), vorgestellt
Das Internet Protokoll bildet die Grundlage des Internets. Das nach dem ISO/OSI-Referenzmodell auf der Vermittlungsschicht [5] arbeitende Protokoll dient zur Adressierung von Geräten bei der Versendung von Datenpaketen und arbeitet dabei verbindungslos. Zur Adressierung werden Netzadressen oder auch IP-Adressen verwendet.
Zurzeit bestehen zwei wesentliche Versionen des Internet Protokolls, IPv4[1] und IPv6[2]. Im IPv4 beträgt die Länge einer IP-Adresse 32 Bit. Durch die zunehmende Verteilung aller 2^{32} möglichen IPv4-Adressen wurde eine Erweiterung des originalen IP-Protokolls eingeführt, das IPv6. Diese führt IP-Adressen mit einer Länge von 128 Bit ein und vergrößert so den Bereich der möglichen IP-Adressen auf 2^{128}.
In beiden Versionen wird die Adresse in einer mit Punkten getrennten Darstellung verwendet. Dabei werden im IPv4 üblicherweise je acht Bit als Dezimalzahl dargestellt und mit Punkten getrennt (beispielsweise `192.168.0.1`). Im IPv6 werden je vier Bit als Hexadezimalzahl, je 16 Bit als Gruppe von vier Hexadezimalzahlen dargestellt und jede Gruppe mit Doppelpunkten voneinander getrennt, Abkürzungen durch Weglassen sind dabei bei wiederholten Nullen möglich (beispielsweise `fe80::`).

[1] https://datatracker.ietf.org/doc/rfc791
[2] https://datatracker.ietf.org/doc/rfc8200/

Zur effizienten Nutzung aller IP-Adressen wurden im Classless Interdomain Routing (CIDR) mit der Einführung einer Netzmaske die Notation für IP-Adressen erweitert. Durch diese Erweiterung werden IP-Adressbereiche, auch Präfix genannt, unter Angabe der Anzahl an Bits ihres Netzanteils mit einem Schrägstrich gekennzeichnet. Bei einer Netzmaske von „`/24`“ werden bei der Adressierung die ersten 24 Bits einer IP-Adresse dem Netzwerk zugewiesen. Der hier beispielhaft aufgeführte IPv4-Adressbereich `10.0.1.0/24` stellt somit die IPv4-Adressen `10.0.1.0` bis `10.0.1.255` dar.
Bei der Weiterleitung von IP-Paketen müssen Router, durch die Einführung der CIDR Notation, in ihrer Routing-Informationstabelle somit nur noch die Netzwerkteil einer IP-Adresse beachten.

2.2. Autonome Systeme

Ein Autonomes System (AS) ist ein Verbund aus Routern unter einer einzigen technischen Administrierung, welche ein gemeinsames inneres Routing-Protokoll und ein gemeinsames Protokoll verwenden, um Pakete außerhalb des Systems zu befördern, wie J.Hawkinson et al. im Request for comment (RFC) 1930 [18] näher spezifiziert. Die Entscheidung darüber, wie ein Paket weitergeleitet wird, nennt man Routing-Policy. Innerhalb eines AS wird dieselbe Routing-Policy angewendet.
Um von anderen Netzwerken erreichbar sein zu können, benötigt ein AS einen eigenen IP-Adressbereich sowie eine einzigartige AS-Nummer. Die Vergabe erfolgt, abhängig vom Standort, von einer Regional Internet Registry (RIR). In Europa ist dies die Réseaux IP Européens Network Coordination Centre (RIPE NCC) [29].
Eine AS-Nummer ist eine 32 Bit lange Nummer zur eindeutigen Identifizierung eines AS. Für die Vergabe hat die Internet Assigned Numbers Authority (IANA) bestimmte AS-Nummer-Bereiche an die jeweiligen RIRs verteilt[3]. Ebenso wurden IP-Adressbereiche, nach der IANA, einer bestimmten RIR zugeteilt[4]. So können der IP-Adressbereich und die AS-Nummer eines AS immer eindeutig einem Kontinent auf dieser Welt zugewiesen werden. Spezielle IP-Adressbereiche (wie beispielsweise `10.0.0.0/8`) und AS-Nummern-Bereiche (wie beispielsweise `64496-64511`) sind dabei der privaten Nutzung oder der Nutzung in lokalen Netzwerken zugewiesen. Die Inter-AS-Kommunikation erfolgt mittels dem Border Gateway Protokoll, welches im nächsten Abschnitt beleuchtet wird.

[3] https://www.iana.org/assignments/as-numbers/as-numbers.xhtml
[4] https://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml

2.3. Border Gateway Protokoll

Das Border Gateway Protokoll (BGP) dient zur Kommunikation zwischen Autonomen Systemen. Das Protokoll wurde 1990 von Lougheed et al. in RFC 1163 [6] erstmalig spezifiziert und dient seit dem als de-facto Standard zu Inter-AS-Kommunikation. Die aktuelle Version ist BGP-4, spezifiziert in RFC 4271 [27]. Die Primäre Funktion des BGP stellt den Austausch von Routing-Informationen mit anderen Systemen dar. Dabei tauschen Router die Erreichbarkeitsinformationen aus, wie sich ein bestimmtes AS erreichen lässt.
Zur Kommunikation über BGP muss ein Router, auch BGP-Sprecher genannt, das Protokoll implementieren. Der Router ist für die Weiterleitung des eigenen IP-Adressbereichs, auch Präfix genannt, sowie für die Weiterleitung der eigenen AS-Nummer zuständig. Man spricht dabei auch vom Annoncieren des Präfixes. Ebenso empfängt der Router die annoncierten Präfixe und dazugehörigen AS-Nummern von anderen Netzwerken.
BGP nutzt das Transmission Control Protokoll (TCP), verwendet den Port 179 und baut eine aktive Sitzung zwischen zwei BGP-Sprechern auf, bei der nach erfolgreichem Zustandekommen der Sitzung periodisch sogenannte ‚Keepalive'-Nachrichten zwischen den BGP-Sprechern ausgetauscht werden. Weiterhin werden Aktualisierungs-Nachrichten (Update Nachrichten) an Sitzungsteilnehmer versendet. Diese beinhalten BGP Informationen, über die die Routen Tabellen eines BGP-Sprechers aktualisiert werden.
Während einer BGP-Sitzung annonciert ein BGP-Sprecher das eigene Präfix an seine Nachbar-AS. Das Nachbar-AS annonciert wiederrum sein Präfix an weitere AS. Dadurch entstehen sogenannte AS-Pfade, eine Verkettung von AS-Nummern, bei der die letzte AS-Nummer das Ausgangs-AS anzeigt.

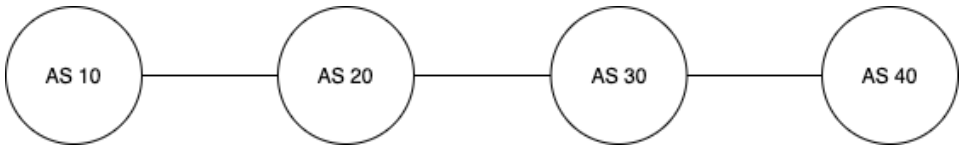

Abbildung 2.1.: AS-Pfad Beispiel

Um exemplarisch einen AS-Pfad darzustellen ist in Abbildung 2.1 eine Topologie mit vier AS dargestellt mit jeweils einer BGP Verbindung zwischen den AS. Die Verbindungsinformationen (AS-Pfad) welche AS40 über AS10 über BGP erhalten würde wäre „30-20-10“, mit dem Ausgangs-AS „AS10“ an letzter Stelle und AS20 und AS30 als Zwischensprünge. Um AS10 Daten zu senden, wüsste AS40 nun, dass es als erstes AS30

kontaktieren müsste. AS30 empfängt die Anfrage von AS40, kennt das Präfix von AS10 und leitet die IP-Pakete an AS20 weiter. Analog passiert dieser Vorgang bei AS20 und schließlich wird das IP-Paket an AS10 gesendet.
Als Pfadvektorprotokoll hat BGP das Ziel, den besten Weg in einem Netzwerk zu finden. Für gewöhnlich erfolgt dies durch die Ermittlung des kürzesten AS-Pfades. Weitere Attribute zur Bestimmung des besten Pfades können je nach Bedarf ebenfalls in Betracht gezogen werden, wie beispielsweise das Attribut „local-preference“, der über einen Integer Wert die Nähe eines Nachbar-AS in einer BGP Sitzung angeben kann. So kann gesteuert werden, ob ein AS eine Peering Verbindung einer Transit Verbindung vorzieht. Die Unterschiede zwischen Transit und Peering werden im Folgenden Abschnitt erläutert

2.4. Transit und Peering

Wie in den vorangegangenen Abschnitten erläutert, tauschen Autonome Systeme Routing-Informationen untereinander aus. Da jedoch das Internet aus verschiedenen Netzwerken besteht, diese Netze unterschiedliche Ziele verfolgen und sich teilweise durch wirtschaftliche Ziele unterscheiden, werden zunächst die Begriffe des Internet Transits und des Internet Peerings erläutert. Der folgende Abschnitt beleuchtet zudem kurz wirtschaftliche Aspekte des Transits und Peerings.

2.4.1. Internet Transit

Für den Zugang zum Internet werden für gewöhnlich Internet Service Provider (ISP) benötigt. Ein Internet Service Provider, zu Deutsch „Internetdienstanbieter“ oder „Internetdienstleister“, bietet Dienstleistungen an, welche sich über das Bereitstellen von Internetanschlüssen hin zu verschiedenen technischen Leistungen erstrecken [23]. Damit der Internetverkehr, von beispielsweise Endverbraucher zu einer Webseite, ermöglicht wird, muss der Internetverkehr durch die verbundenen Netze fließen. Der ISP wird für den Internetanschluss von seinen Kunden bezahlt. Der ISP wiederrum muss andere Betreiber (also nächstgrößere ISPs) bezahlen, um den Internetverkehr seiner Kunden weiterzuleiten. Dies wird im Allgemeinen Internet Transit genannt.
William B. Norton beschreibt in seinem Buch „The Internet Peering Playbook“ [23] Internet Transit als ein Geschäftsmodell, welches zwei Hauptfunktionen erfüllt: (a) Internet Routing-Informationen an Kunden annoncieren, und (b) Kunden Routing-Informationen an das restliche Internet annoncieren.

Im Fall (a) teilt der Transit Anbieter seinen Kunden die Routing-Informationen der AS mit, welche der Anbieter erreichen kann. Im Fall (b) wird die Information des Kunden-AS über den Transit Anbieter an alle anderen AS weitergeleitet, die der Anbieter erreicht. ISPs können regional, über-regional bis hin zu international agieren. Sie kommen in verschiedenen Netzwerkklassen vor, auch Tier-Klassen genannt. Tier-Klassen unterscheiden sich darin, ob ein Netzwerk Internet Transit (bei einem anderen Netzwerk) erwirbt, oder Internet Transit an ein anderes Netzwerk verkauft.
Verkauft ein Netzwerk Internet Transit, dient so als Transit Anbieter für andere Netzwerke, erwirbt selbst jedoch keinen Transit, so wird es als Tier-1 Netzwerk eingestuft. Tier-1 Netzwerke betreiben untereinander Peering und tauschen Erreichbarkeitsinformationen aus, dadurch besitzen sie die totale Erreichbarkeit. Da Tier-1 Netzwerke meist über-regional bis international agieren, werden sie als die größte Netzwerkklasse eingestuft. Tier-2 Netzwerke erwerben Transit bei größeren Netzwerken und bieten Transit an kleinere Netzwerke. Diese sind meist als über-regionale bis regionale Netzwerk-Betreiber vertreten. Tier-3 Netzwerke erwerben Transit bei größeren Netzwerken, verkaufen jedoch keinen Transit. Diese Netzwerke bieten kleineren AS (meistens Unternehmen oder Einzelpersonen) Internetzugänge an.
William B. Norton beschreibt [23] wirtschaftliche Implikationen, welche durch die hierarchischen Strukturen im Transit-Geschäft entstehen. Es entstehen Abhängigkeiten, von den Kunden gegenüber den Transit Anbietern. Das Wegfallen einer Transit Verbindung und der Routing-Annoncierung hätte für einen ISP zur Folge, seinen Kunden keinen Internetzugang mehr anbieten zu können. Aufgrund dessen beziehen ISPs oft Transit von mehreren Anbietern, entweder aus wirtschaftlichen Gründen, um eine Ausfallsicherheit zu gewährleisten oder zur Last-Verteilung. Dies wird im Allgemeinen als ‚Multi-Homing' bezeichnet. Multi-Homing stellt für ISPs zudem eine Möglichkeit dar, den Internetverkehr der Kunden genauer zu routen und so entscheiden zu können, wie der Internetverkehr versendet wird, da ein Internet Transit Anbieter den Internetverkehr zumeist pro übertragene Dateneinheit (beispielsweise Euro pro Megabyte pro Sekunde) abrechnet. Um dieser Abhängigkeit entgegenzuwirken, haben ISPs zusätzlich die Möglichkeit, sich mit weiteren ISPs mittels Peering zusammenzuschalten.

2.4.2. Internet Peering

Internet Peering bezeichnet die Geschäftsbeziehung, bei der sich zwei Parteien gegenseitig Zugang zu den Kunden des jeweils anderen verschaffen [23]. Peering ist meistens eine abrechnungsfreie Beziehung, sodass keine Partei die jeweils andere für den Zugang

bezahlt. Beim Peering werden die Routing-Informationen, über das jeweils eigene Netz, zwischen den beiden Parteien ausgetauscht.
Beim Peering wird zwischen Public Peering (zu Deutsch: Öffentliches Peering), Private Peering (zu Deutsch: Privates Peering) und Paid Peering (zu Deutsch: Bezahltes Peering) unterschieden. Das Public Peering bezeichnet das Peering über ein öffentlich geteiltes Medium, wie beispielsweise ein Ethernet Switch (Internet-Knoten). Beim Private Peering wird eine dedizierte Layer 2 Schaltung über eine direkte Verbindung oder ein virtuelles LAN (VLAN) an einem Internet-Knoten realisiert. Beim Paid Peering wird für das Zustandekommen der Peering-Beziehung eine Kompensation in Form von Bezahlung oder einem sonstigen Tausch angeboten. Tier-1-Netzwerke schließen sich beispielsweise mittels Private Peering untereinander zusammen und bilden so den „Kern des Internets“ [23]. Da an Internet-Knoten zumeist öffentliches Peering zustande kommt, wird sich vertiefend mit der öffentlichen und abrechnungsfreien Art des Peerings, dem Public Peering, beschäftigt.
Durch Peering erhalten beide Peering-Partner, die teilnehmenden AS, die Routing-Informationen des jeweils anderen. Der Datenaustausch erfolgt direkt oder indirekt über ein Medium. Die Erreichbarkeitsinformationen beschränken sich dabei ausschließlich auf die Informationen über das AS des Peering-Partners. Dies stellt einen Unterschied zum Internet Transit dar, da der Transit Anbieter die Routing-Informationen aller erreichbaren Netze mit seinen Kunden teilt.
Da Public Peering an Internet-Knoten erfolgt, wird sich im nachfolgenden Abschnitt vertiefend mit den Aspekten des Peerings an Internet-Knoten und Internet-Knoten im Allgemeinen auseinandergesetzt.

2.5. Internet-Knoten

Internet-Knoten, im Englischen Internet Exchange Point (IXP), sind technische Einrichtungen, welche den Zweck der Zusammenschaltung von Netzwerken verfolgen. Nach der von der European Internet Exchange Association (Euro-IX) anerkannten Definition der Internet Exchange Federations (IX-F) ist ein Internet-Knoten eine Netzwerkeinrichtung, welche die Verbindung und den Datenaustausch zwischen mehr als zwei Autonomen Systemen ermöglicht. Die Definition spezifiziert weiterhin, dass nur AS an einem Internet-Knoten teilnehmen und dass der Internetverkehr zwischen zwei AS nicht verändert oder auf sonstige Art und Weise gestört werden sollte [13] [20].
Internet-Knoten verbinden bis zu tausende AS, verkürzen die Distanzen zwischen den

AS und verringern die Latenz bei der Übertragungszeit, indem der Internetverkehr lokal gehalten wird [30]. Lokal bedeutet hierbei geografisch gesehen, dass der Internetverkehr, wenn möglich, über möglichst kurze Wege geleitet wird und der ‚Umweg' über einen Transit vermieden wird.
Für Public Peering stellen Internet-Knoten Anschlüsse gegen Entgelt zur Verfügung. Im Gegensatz zum Internet Transit verlangen (kommerzielle) Internet-Knoten einen festen monatlichen Betrag für einen Anschluss mit einer maximalen Datenraten-Obergrenze, beispielsweise einen Ethernet-Anschluss mit einer Kapazität von einem Gigabit pro Sekunde. Ökonomisch betrachtet stellt dies eine Alternative zum Internet Transit dar, da statt eines variablen Betrages, mit einem festen monatlichen Betrag eine Vielzahl von Netzwerken an einem Internet-Knoten erreicht werden können, sofern sie öffentliches Peering betreiben.
Betrachtet werden im Folgenden die drei größten europäischen Internet-Knoten, gemessen am Datendurchsatz und an den verbundenen Netzwerken: AMS-IX, in Amsterdam, DE-CIX in Frankfurt am Main, und LINX in London [2] [8] [21].

Tabelle 2.1.: Europäische Internet-Knoten im Vergleich

Internet-Knoten	Verbundene AS	Max. Datendurchsatz
AMS-IX	879	10.559 Tb/s
DE-CIX	2.465	10.799 Tb/s
LINX	999	6.511 Tb/s

Quelle: [9] [3] [22]

Tabelle 2.1 zeigt eine Übersicht der jeweils verbundenen AS sowie des maximalen Datendurchsatzes. Die in Tabelle 2.1 angegebenen Daten stammen von den Betreibern der Internet-Knoten selbst. Die Anzahl der verbundenen AS bezieht sich auf die gesamte Plattform des jeweiligen Internet-Knotens. Der angegebene Datendurchsatz für AMS-IX und LINX bezieht sich auf den aggregierten Datendurchsatz auf der jeweiligen Plattform. Der angegebene Datendurchsatz für DE-CIX bezieht sich auf den Standort in Frankfurt am Main.
Einige Publikationen stellten den Datendurchsatz von Internet-Knoten bereits mit Internet Transit Anbietern gleich. So haben Ager et. al in [1] gezeigt, dass bereits 2014 große europäische Internet-Knoten ähnliche Datenmengen pro Monat transportieren wie Internet Transit Anbieter. Der gemessene maximale Datendurchsatz von 10.799 Tb/s (Terrabit pro Sekunde) bei DE-CIX würde circa einem Tages-Datendurchsatz von rund 116,7 Petabyte entsprechen. Im Vergleich zu den im Jahr 2014 aufgeführten Datenmen-

gen in [1] von 14 Petabyte pro Tag stellt dies eine große Steigerung dar. Die Menge des Datendurchsatzes von Internet Transit Anbietern im Jahr 2021 lässt sich nicht genau bestimmen. Mit der wachsenden Anzahl an Teilnehmern sowie steigendem Datendurchsatz pro Jahr, welcher durch Internet-Knoten fließt, lässt sich allerdings eine gewisse Relevanz für das Internet ableiten. So hat sich beispielsweise der Datendurchsatz am DE-CIX Internet-Knoten im Vergleich zum Datendurchsatz vor fünf Jahren verdoppelt [9].
Um diese Datenmengen ausfallsicher und zuverlässig transportieren zu können, werden bei DE-CIX redundante Verbindungen und Systeme eingesetzt, welche eine Verfügbarkeit von 99,99% für einen Peering Service bieten [10]. Um eine möglichst skalierbare Verteilung der Informationen und Verbindung der Teilnehmer zu erreichen, werden an Internet-Knoten zumeist Routen-Server eingesetzt [28]. Routen-Server dienen zur Abwicklung der BGP-Sitzungen aller Teilnehmer an Internet-Knoten. Diese verteilen die Routen Annoncierungen aller Teilnehmer und dienen als Verbindungspunkt für den Internet-Knoten. In ihrem Ratgeber über ihren Routen-Server [11] zeigen die Entwickler von DE-CIX die Prozesse, die einer Annoncierung der Route vorausgehen. So werden beispielsweise bestimmte Präfixe herausgefiltert, mögliche Angriffe (falsche BGP Annoncierungen) erkannt und Routen Authentifizierungen durchgeführt.
An diesen Prozessen kann abgeleitet werden, dass aus technischer Sicht gesehen, ein großer Aufwand betrieben wird, die Routen Informationen an Internet-Knoten valide zu halten und so eine möglichst große Sicherheit beim Austausch von Informationen über BGP zu bieten. Für die Ausfallsicherheit wird ein hoher Aufwand betrieben, denn der Ausfall eines wichtigen Systems könnte das Wegfallen der Internetkonnektivität einer gesamten Region bedeuten[5].

[5]https://www.heise.de/newsticker/meldung/DE-CIX-Stromausfall-legte-Internet-lahm-4014125.html

3. Stand der Technik

In diesem Kapitel werden bereits bestehende Lösungen und Technologien zur Implementierung eines Internet-Knotens zu Demonstrationszwecken näher betrachtet. Zunächst wird ein Projekt vorgestellt. Im Anschluss wird die in dieser Bachelorthesis eingesetzte Hardware und Software beschrieben.

3.1. Mini-Internet

Formen eines praktischen Kurses in Lehrveranstaltungen, um die Konzepte des Routings und die Kommunikation zwischen Autonomen Systemen zu veranschaulichen, existieren bereits. Im Folgenden wird das „Mini-Internet“ Projekt[1] der ETH Zürich vorgestellt. Die Entwickler des Projektes geben in ihrem technischen Bericht [19] einen Einblick in die Implementierung und erklären dabei wie ihr Projekt einen theoretischen Kurs um eine praktische Komponente erweitern kann.
Die Entwickler geben an, es handele sich dabei um einen interaktiven Kurs, bei dem die Teilnehmer aktiv Autonome Systeme mit Hilfe einer Routing-Software zusammenschalten und so die Konzepte der Konfiguration, Design und Überwachung von Netzwerken lernen und verstehen können. Weiterhin sollen die Konzepte des Peerings, des Transits und des BGPs in dem Kurs nähergebracht werden.
Der Kurs basiert dabei auf dem Konzept einer virtuellen Umgebung, bei dem die Teilnehmer per Fernzugriff, einem Virtual Private Network, zu Deutsch: Virtuellen Privaten Netzwerk (VPN), beitreten und mittels Docker[2], einer Technologie zur Containerisierung, einzelne Netzwerk-Komponenten erzeugen. Laut den Entwicklern können die Betreiber des Mini-Internet zuvor eine Topologie aus Netzwerk-Komponenten wie Switche und Routern mittels Open vSwitch[3], eine freie Software zur Virtualisierung von Switchen, und FRRouting zur Steuerung von Routern erzeugen und so den Kurs ihren Bedürfnissen anpassen. Einen Internet-Knoten zur Zusammenschaltung der von den

[1] https://mini-inter.net
[2] https://www.docker.com
[3] https://www.openvswitch.org

Teilnehmern erzeugten AS werde von den Betreibern ebenfalls erzeugt. Im Bericht werden Konzepte zur Überwachung des erzeugten Netzwerks sowie Aufgaben, welche die Teilnehmer erfüllen sollen, erläutert.
Es zeigt sich in diesem Bericht ein Konzept eines vollwertigen praktischen Kurses, welche eine aktive Teilnahme der Studenten und einen hohen Aufwand seitens der Entwickler erfordere. Betrieben wird, laut den Entwicklern, das „Mini-Internet“ auf einem Server mit 24 Prozessor Kernen sowie 256 Gigabyte Arbeitsspeicher, dies solle bei einer Teilnehmer-Anzahl von 100 Studenten ausreichen. Die Entwickler des Mini-Internet Projekts geben an, eine Topologie bräuchte bis zu zwölf Stunden, bis sie erstellt und einsatzbereit wäre. Davon lässt sich ableiten, dass es einen hohen Aufwand an Konfigurations-Leistung erfordern könnte sowie einen Server, welcher die nötige Rechen-Leistung erbringt. Da ein Server mit solchen Hardwareanforderungen allerdings im Preissegment von mehreren tausend Euros liegt, wird sich im Folgenden mit kostengünstigeren Einplatinencomputern beschäftigt.

3.2. Raspberry Pi

Der Raspberry Pi ist ein von der Raspberry Pi Foundation entwickeltes Einplatinencomputer System. Es findet seinen Ursprung an der Cambridge Universität, wo es als kostengünstiges System für Schüler und Studenten konzipiert wurde [12].
Der Einplatinencomputer enthält ein Ein-Chip-System sowie einen ARM-Prozessor. Der bis zu ca. 9 x 6 cm große Raspberry Pi verfügt über verschiedene Anschlüsse, darunter beispielsweise USB und HDMI sowie einige Schnittstellen wie Bluetooth, WLAN und Ethernet und kann je nach Modelausführung vielseitig eingesetzt werden. Durch die Verbindung von Peripherie-Geräten, wie Tastatur und Maus sowie einem Monitor eignet sich der Raspberry Pi Modell B ebenso als Computer, wie auch als Server durch die Kommunikation per Kommandozeile via Fernzugriff. Das Laden des Betriebssystems erfolgt dabei mittels einer SD-Karte (im Englischen Secure Digital). Als offizielles unterstützendes Betriebssystem führen die Hersteller das Raspberry Pi OS auf. Weitere Linux basierte Betriebssysteme werden ebenfalls unterstützt [26].
Zum Stand dieser Bachelorthesis existieren 13 Versionen des Gerätes. Dabei ist die aktuelle Ausführung der „Raspberry Pi Model 4 B“, erhältlich mit verschieden großen Arbeitsspeichern. Preislich liegt das Modell 4 B, je nach Version, zwischen 40 und 100 Euro. Seit dem Verkaufsstart des Raspberry Pi Model B 2012 wurden bis Ende 2020 über 37 Millionen Geräte verkauft [14].

Im Vergleich zu herkömmlichen Geräten, welche Anwendung in Rechenzentren finden, bietet der Raspberry Pi dadurch bei einem geringen Anschaffungspreis sowie einer geringen Betriebsspannung von 5 V und bei max. 7 W Leistung eine kostengünstige Alternative. Zusätzlich kommen die Raspberry Pi Einplatinencomputer ohne zusätzliche Kühlvorrichtungen aus, die, während dem Betrieb, Geräusche verursachen. Dadurch sind die Einplatinencomputer ideale Kandidaten für Lehrveranstaltungen und Vorführungen. Die Kommunikation im Versuchsaufbau erfolgt mittels eines Ethernet-Anschlusses. Im Folgenden werden daher die aktuelle Ausführung des Raspberry Pi, das Model 4 (Abbildung 3.1) sowie dessen Vorgänger, dem Model 3 B+ (Abbildung 3.2) näher betrachtet. Tabelle 3.1 führt die technischen Daten der beiden Modelle auf.

Abbildung 3.1.: Raspberry Pi Model 4 B
Quelle: https://www.raspberrypi.com/products/

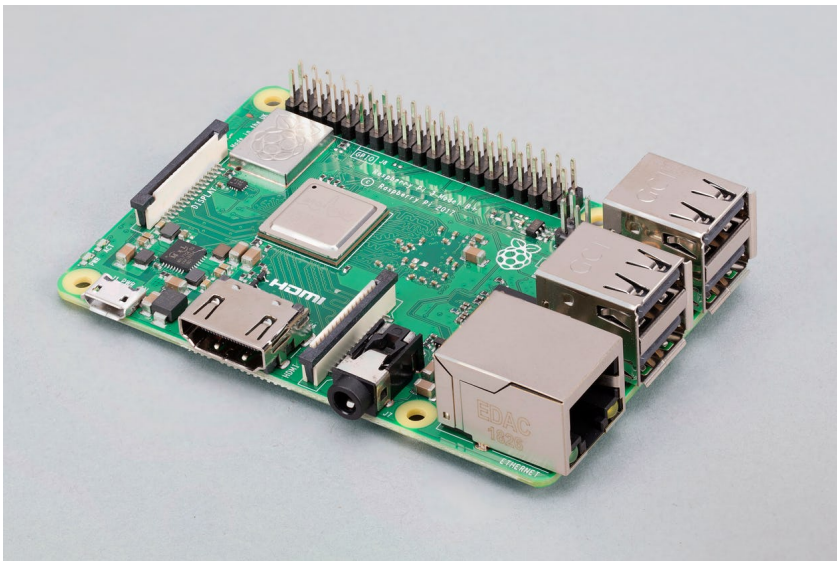

Abbildung 3.2.: Raspberry Pi Model 3 B+
Quelle: https://www.raspberrypi.com/products/

Tabelle 3.1.: Technische Daten Raspberry Pi Model 3 B+/ Model 4 B

Modell	Raspberry Pi Model 4 B	Raspberry Pi Model 3 B+
Prozessor	ARM Cortex-A72 1500 MHz	ARM Cortex-A53 1400 MHz
Ein-Chip-System	Broadcom BCM2711B0	Broadcom BCM2837B0
Arbeitsspeicher	4096 MB LPDDR4-SDRAM	1024 MB LPDDR2-SDRAM
Netzwerk	Ethernet: 1000 Mbit/s WLAN: 802.11.b/g/n/ac	Ethernet: 1000 Mbit/s (über USB) WLAN: 802.11.b/g/n/ac
Anschlüsse	2x Micro-HDMI 2x USB 2.0 2x USB 3.0	1x HDMI 4x USB 2.0
Betriebsspannung	5,0 V	5,0 V

Quelle: https://www.raspberrypi.com/

3.3. Orange Pi

Orange Pi ist ein quelloffenes Einplatinencomputer-Projekt, ähnlich zum Raspberry Pi der Raspberry Pi Foundation. [24] Vertrieben wird der Orange Pi von der Shenzhen Xunlong Software CO. Limited.
Die Orange Pi Plattform ist in vielen Aspekten ähnlich zu der Plattform des Raspberry Pi. Es basiert ebenfalls auf einem Ein-Chip-System unterstützt von einem ARM-Prozessor, ist ebenso bis zu ca. 9 x 6 cm groß und enthält, je nach Ausführung und Modell, viele verschiedene Anschlüsse wie USB, HDMI und Ethernet.
Der Orange Pi ist in vielen Ausführungen erhältlich. Dabei existieren Einplatinencomputer, welche ähnliche technische Daten aufweisen, wie die Geräte der Raspberry Pi Foundation. Es existieren auch weitaus kleinere Computer. Der 2017 erschienene Orange Pi Zero Plus ist ein Einplatinencomputer, welcher mit seiner Größe von 4,6 x 4,8 cm weitaus kleiner ist, als ein Raspberry Pi Model 4 B. Preislich liegt das Model Zero Plus bei circa 25 Euro[4]. Das Gerät ist in Abbildung 3.3 abgebildet.

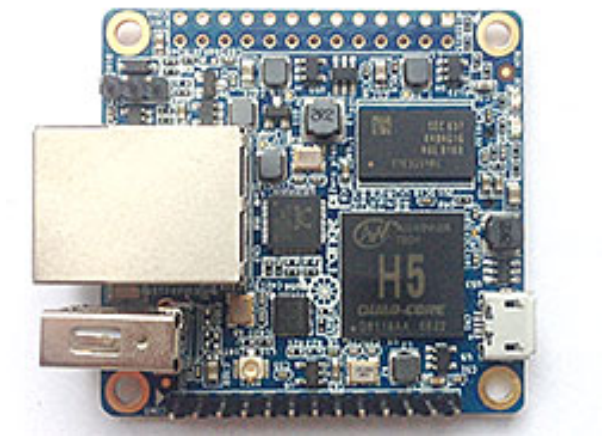

Abbildung 3.3.: Orange Pi Zero Plus
Quelle: https://linux-sunxi.org/Xunlong_Orange_Pi_Zero_Plus

Trotz der geringeren Größe enthält der Orange Pi Zero Plus einen Ethernet-Port mit einem Gigabit pro Sekunde Kapazität sowie eine WLAN-Antenne und einem USB 2.0 Anschluss. Das Betriebssystem wird, wie die meisten Einplatinencomputer, über eine SD-Karte geladen. Weitere technische Daten sind in Tabelle 3.2 aufgeführt. Die Orange Pi Einplatinencomputer kommen ebenfalls ohne zusätzliche Kühl-Vorrichtungen aus und sind während des Betriebs lautlos, im Vergleich zu herkömmlichen Servern.

[4]https://www.aliexpress.com/item/1005001688471438.html

Tabelle 3.2.: Technische Daten Orange Pi Zero Plus

Modell	Orange Pi Zero Plus
Prozessor	ARM Cortex-A53 1200 MHz
Ein-Chip-System	Allwinner H5
Arbeitsspeicher	512 MB DDR3
Netzwerk	Ethernet: 1000 Mbit/s WLAN: 802.11.b/g/n
Anschlüsse	1x USB 2.0
Betriebsspannung	5,0 V

Quelle: http://www.orangepi.org/OrangePiZeroPlus/

3.4. FRRouting

Free Range Routing oder FRRouting ist eine kostenlose und quelloffene Routing-Software der Linux Foundation für Linux und Unix Plattformen, welche eine Vielzahl von Routing-Protokollen implementiert [15]. FRRouting basiert auf der Routing-Software Quagga [25], welches wiederrum auf der Routing-Software Zebra [31] aufbaut und wurde von den FRRouting Entwicklern weiterentwickelt.
FRRouting bietet verschiedene IP-Routing Dienste an, die jeweils ein Routing-Protokoll implementieren. Darunter BGP, Open Shortest Path First (OSPF), Intermediate System to Intermediate System (IS-IS) und viele mehr. Dabei emuliert FRRouting einen Router, indem es Routing-Entscheidungen in den Betriebssystem Kernel lädt und so dem Netzwerk Kernel des Betriebssystems die Möglichkeit gibt Pakete entsprechend weiterzuleiten.
Die Entwickler von FRRouting bieten zudem auf Ihrer Webseite eine ausführliche Dokumentation zur Installation und Konfiguration von einer Vielzahl von Anwendungsfällen.

Die Hardwareanforderungen von FRRouting bleiben, laut den Entwicklern, auf einem minimalen Niveau. Die Menge an Routing-Informationen, die von den Geräten verarbeitet werden sollen und die Anwendungsfälle seien dabei entscheidend. Nach Angabe der Entwickler eigne sich FRRouting speziell für einen Versuchsaufbau mit einigen Einplatinencomputern. Es biete sogar die Möglichkeit als Internet-Knoten Kern-Router zu fungieren, wenn man genügend Ressourcen zur Verfügung stellt [17].

3.5. Netzwerk-Kommandos

Traditionelle Kommandos zur Messung von Netzwerk-Konnektivität und zur Konfiguration von Netzwerk Schnittstellen sind bereits in vielen Betriebssystemen enthalten. Wie auch die Entwickler des Mini-Internet Projekts der ETH Zürich wird in dieser Bachelorthesis durch die Verwendung von solchen Kommandos die Funktionalität des Versuchsaufbaus verifiziert. Dieser Abschnitt stellt die Kommandos `ping`, `traceroute`, `ip` und `iperf` vor. Diese Linux-basierten Kommandos bieten eine breite Bandbreite an Funktionen und können mit vielseitigen Parametern beeinflusst werden [5].

3.5.1. ping

Das auf dem Internet Control Message Protocol (ICMP) basierendes Diagnosewerkzeug `ping` dient zur Messung der Erreichbarkeit von Zielsystemen. Dabei werden ICMP Echo-Anfragen an das entsprechende Zielsystem gesendet. Als Parameter wird das Zielsystem als IP-Adresse oder als Hostnamen übergeben. Weitere Parameter können beispielsweise die Anzahl an Anfragen oder die zu verwendende Schnittstelle sein. Als Rückgabe wird die benötigte Zeit zum Zielsystem pro Echo-Anfrage ausgegeben.
Das folgende Kommando sendet fünf Anfragen an die Zieladresse `www.example.com`:
`ping -c 5 www.example.com`

3.5.2. traceroute

Mit `traceroute` wird der Weg ermittelt, den ein Datenpaket zu einem Zielsystem zurücklegt. Dies erfolgt durch die Ermittlung der Router, die die versendeten ICMP Echo-Anfragen bis zum Ziel vermitteln. Als Ziel wird, wie bei `ping` ebenfalls, eine IP-Adresse oder ein Hostnamen als Parameter übergeben. Als Rückgabe werden die IP-Adressen und die Namen der Router auf dem zurückgelegten Weg ausgegeben. Das folgende Kommando ermittelt den Weg vom lokalen Router zum Zielsystem `www.example.com` über die Schnittstelle `wlan0`:
`traceroute www.example.com -i wlan0`

3.5.3. ip

Zur Ausgabe und zur Modifizierung von Routing-Geräten, Netzwerkgeräten und Schnittstellen kommt das Kommando `ip` zum Einsatz. Es fasst einige klassische Kommandos,

darunter `route` (Routing Tabelle) und `ifconfig` (Schnittstellen), unter einem neuen Kommando zusammen und bietet so vielseitige Funktionen. Das folgende Beispiel weist der Schnittstelle `eth0` die IP-Adresse `10.0.4.10/24` zu:

```
ip address add 10.0.4.10/24 dev eth0
```

Eine weitere Funktion von ip zeigt das folgende Beispiel, bei dem die IP-Adresse `192.168.0.1` als Standardroute über die Schnittstelle `wlan0` festgelegt wird:

```
ip route add default via 192.168.0.1 dev wlan0
```

3.5.4. iperf

Die freie Software `iperf` dient zur Ermittlung des Datendurchsatzes eines Netzwerkes. Es führt standardmäßig eine Messung der maximal erreichbaren Bandbreite mit Verbindungsprotokollen wie TCP/UDP durch. Es können Parameter, wie die Verwendung eines bestimmten Protokolls oder Einstellungen in Bezug auf zeitliche Koordinierung und Puffer übermittelt werden. Für jeden Test wird die Bandbreite, versendete Datenmenge sowie eventueller Datenverlust und andere Parameter ausgegeben. Das folgende Kommando testet die Bandbreite des Systems `192.168.0.50` und sendet mittels TCP Pakete an das Zielsystem:

```
iperf -c 192.168.0.50
```

Voraussetzung ist dabei, dass das Zielsystem (`192.168.0.50`) den `iperf` Dienst bereitstellt und auf die Anfrage lauscht.

3.5.5. systemctl

Mit dem Kommandozeilenwerkzeug `systemctl` können Befehle an `systemd` gesendet werden und so Dienste verwaltet werden. `systemd` ist ein System- und Sitzungs-Verwaltungsdienst, welches für die Verwaltung aller laufenden Dienste auf einem System zuständig ist. Mit `systemctl` können Dienste unter anderem gestartet, neu geladen oder beendet werden. Das folgende Beispiel bewirkt das automatische Starten (`enable`) des Dienstes `dhcpcd`:

```
systemctl enable dhcpcd
```

4. Design

In diesem Kapitel wird das gewählte Design beschrieben, um in einem Testszenario einige zusammengeschaltete Autonome Systeme an einen Internet-Knoten zu simulieren. Im ersten Abschnitt werden die Anforderungen an den Versuchsaufbau beschrieben. Dabei wird die eingesetzte Hardware im Kontext zur Fragestellung dieser Bachelorthesis gesetzt sowie Designentscheidungen im Versuchsaufbau begründet. Im Anschluss wird die eingesetzte Hardware beschrieben.

4.1. Anforderungen an den Versuchsaufbau

In diesem Abschnitt wird eine Übersicht über die Anforderungen an das Testszenario gegeben. Es werden benötigte Komponenten sowie benötigte Software-Konzepte erläutert.

Um in einer Lehrveranstaltung, einem praktischen Kurs oder einer Vorführung einen Internet-Knoten zu simulieren, werden zunächst die Definitionen aus Kapitel zwei für einen Internet-Knoten betrachtet. Die Definition nennt eine mindest Teilnehmeranzahl von drei („mehr als zwei“) AS an einem Internet-Knoten. Somit wird ein Versuchsaufbau von mindestens drei bis mehr AS in Betracht gezogen. Um ein AS abbilden zu können, werden eigene IP-Adressräume sowie eigene AS-Nummern für die teilnehmenden AS benötigt, wie in der Definition für Autonome Systeme in Kapitel zwei erläutert.
Da ein Autonomes System einen Verbund aus Routern beschreibt, können mehrere Router oder auch ein einziger BGP-fähiger Router zur Kommunikation verwendet werden. Die Router innerhalb eines AS können hierbei mittels Software simuliert werden. Die Inter-AS-Kommunikation erfolgt mittels BGP. Daher wird ein Router benötigt, der BGP implementiert. Die Implementierung kann dabei über FRRouting erfolgen. FRRouting bietet BGP Funktionen sowie die Möglichkeit eigene Konfigurationen, wie beispielsweise eigene Topologien, vorzunehmen.
Die Definition des IX-F von Internet-Knoten lässt Spielraum für eine Interpretation bei

der Umsetzung einer Netzwerkeinrichtung. So kann ein Switch, der auf der Sicherungsschicht operiert, genutzt werden, um mehrere Router zu verbinden. Der Switch sollte dabei mindestens drei Schnittstellen verbinden können. Die Kommunikation zwischen den AS sollte, wie in der Definition des IX-F beschrieben, ohne Störung und ohne Nutzung eines dritten AS erfolgen.
Da der Versuchsaufbau Anwendung in einem Vortrag oder einer Lehrveranstaltung findet, sollte der Versuchsaufbau möglichst mobil und transportabel sein, hinreichend lautlos, automatisierbar, die Möglichkeit einer Visualisierung und Demonstration bieten sowie den Ablauf der Veranstaltung nach Möglichkeit nicht behindern.
Die Wahl fiel daher auf die Nutzung von Einplatinencomputern und einem Ethernet Switch. Die Einplatinencomputer eignen sich ideal durch ihren mobilen Charakter, welcher durch ihre Größe hervorgerufen wird sowie durch ihre niedrigen Energie- und Kühlungs-Anforderungen für einen Aufbau in einer Lehrveranstaltung. Weiterhin bieten die meisten Einplatinencomputer Schnittstellen zur Netzwerk-Kommunikation über Ethernet oder WLAN sowie Anschlüsse für Peripherie und Monitore. Dadurch wäre es möglich, den Versuchsaufbau zu visualisieren und Router zu emulieren. Einplatinencomputer können vielseitig konfiguriert werden. Sie benötigen lediglich ein Betriebssystem, welches für die jeweilige Hardware kompatibel ist. Durch den SD-Karteneinschub, der in vielen Einplatinencomputern verbaut wird, können verschiedene Betriebssysteme beziehungsweise Konstellationen mit verschiedenen Betriebssystemen untersucht werden.
Spezielle BGP-fähige Router, die für die Anwendung in Rechenzentren konzipiert werden oder Anwendung im professionellen Bereich finden, sind meistens auf die Spezifikationen des Herstellers angewiesen. So fordern sie, je nach Hersteller, ein eigenes Betriebssystem und benötigen eine eigene Konfigurations-Syntax. Weiterhin bieten sie nicht die Flexibilität und Mobilität, die im Gegensatz Einplatinencomputer bieten können.
Durch den Einsatz von Router-Software wie FRRouting wird die Kommunikation der Router in der realen Welt abgebildet. FRRouting bietet die Möglichkeit zur Inter-AS-Kommunikation über einen BGP-Dienst. Ein weiterer Vorteil ist die flexible Anpassung von Routing-Konstellationen und Testszenarien.

4.2. Topologie

Dieser Abschnitt behandelt die gewählte Topologie des Versuchsaufbaus, welche aus den Anforderungen des vorhergien Abschnitts hervorgegangen ist. Bei der Designentscheidung wurden ebenfalls die in Kapitel zwei behandelten Konzepte zu Autonomen Systemen, BGP sowie zu Peering und Transit berücksichtigt.

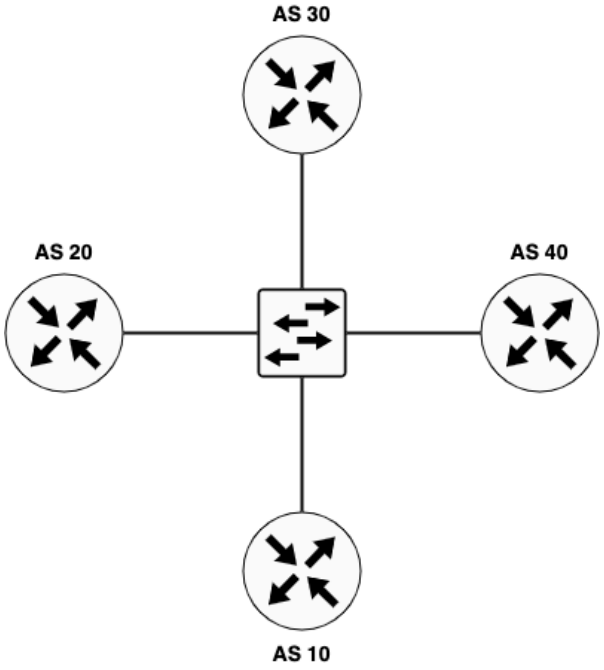

Abbildung 4.1.: Versuchsaufbau Topologie

Der Versuchsaufbau zur Simulation eines Internet-Knotens wird über vier Router realisiert, welche mittels BGP über ein gemeinsames Medium kommunizieren können. In Abbildung 4.1 ist der schematische Aufbau dargestellt. Dabei sind Autonome Systeme als Router-Objekt symbolisch zusammengefasst und stellen jeweils ein vollwertiges AS dar. Das Netzwerk bildet eine Stern-Topologie mit vier AS, die als teilnehmende Netzwerke fungierten, sowie einem Switch im Zentrum, welcher alle vier AS verbindet. Die Vergabe der AS-Nummern fiel, aus Gründen der Leserlichkeit und der Verständlichkeit, auf die Nummern 10 bis 40. Laut der American Registry for Internet Numbers[1], sind diese AS-Nummern zwar bereits vergeben, da die Netzwerke aber niemals „nach außen“, also in das Internet, kommunizieren, stellt die Vergabe der Nummern in einem lokalen

[1] https://whois.arin.net

Netzwerk kein Problem dar.
Durch diese Topologie können alle verbundenen Netze direkt miteinander kommunizieren, ohne ein AS als Internet Transit Anbieter benötigen zu müssen. Diese Topologie stellt einen Internet-Knoten nach der in Kapitel zwei beschriebenen Definition und nach den in Abschnitt 4.1 beschriebenen Anforderungen dar. Der zentrale Switch operiert dabei auf der Sicherungsschicht und dient einzig zur Weiterleitung der Datenpakete, bleibt somit unsichtbar für die verbundenen AS und stört oder verändert die Kommunikation zwischen den AS nicht.

Tabelle 4.1.: AS IP-Bereich und IP-Adressen

AS	10	20	30	40
IP-Bereich	10.0.1.0/24	10.0.2.0/24	10.0.3.0/24	10.0.4.0/24
Schnittstelle IP-Adresse	eth0 10.0.1.1	eth0 10.0.2.1	eth0 10.0.3.1	eth0 10.0.4.1
Schnittstelle IP-Adresse	eth0:0 10.0.3.10	eth0:0 10.0.1.20	eth0:0 10.0.2.30	eth0:0 10.0.3.40
Schnittstelle IP-Adresse	eth0:1 10.0.4.10	eth0:1 10.0.4.20		

Tabelle 4.1 stellt die Zuordnung von den gewählten AS Nummern im Versuchsaufbau zum gewählten IP-Adressbereich der AS (kurz: IP-Bereich) und die IP-Adressen der Schnittstellen am jeweiligen AS dar. Aus Gründen der Leserlichkeit sind die kürzeren IPv4-Adressen ausgewählt worden, statt IPv6-Adressen.
Der IP-Bereich der AS ist so gewählt, dass die Router über einen hinreichend großen Bereich zum Adressieren von Geräten verfügen. Jedes AS erhält einen IP-Bereich von „10.0.X.0/24“. Das „X“ stellt hierbei die erste Ziffer der AS-Nummer dar. Eine Netzmaske von 255.255.255.0 („/24“) stellt bis zu 256 IP-Adressen pro AS zur Verfügung.
Die IP-Adressen der Schnittstellen werden statisch gesetzt. Dabei stellt das dritte Oktett der IP-Adresse das Ziel-Netzwerk dar, das vierte Oktett die AS-Nummer des Netzwerkes, welches die Schnittstelle implementiert. Dadurch kann die folgende Netzwerk Topologie umgesetzt werden.

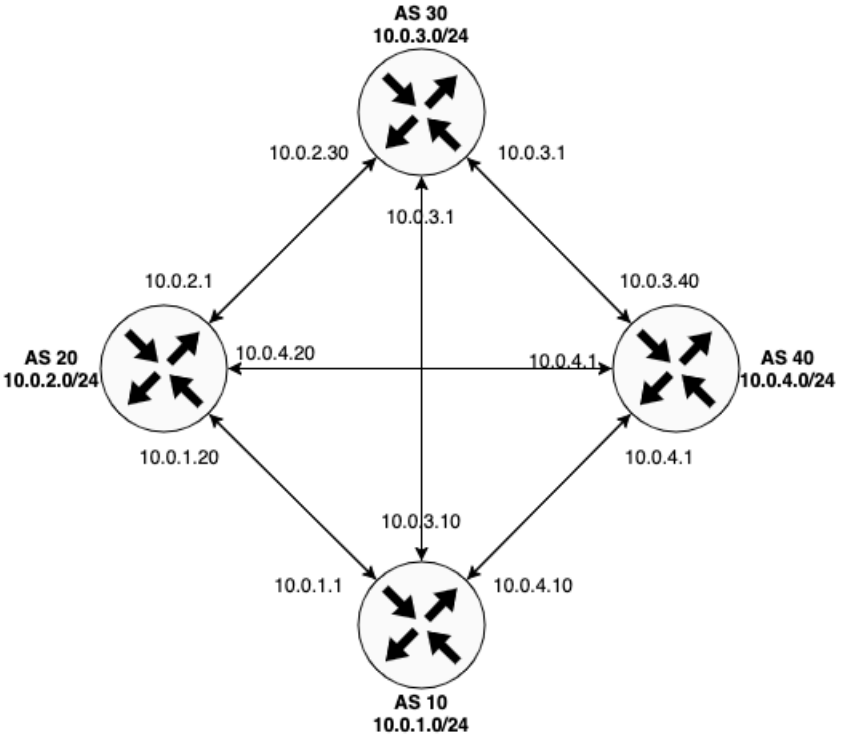

Abbildung 4.2.: Netzwerk Topologie

Abbildung 4.2 stellt die Netzwerk Topologie des Versuchsaufbaus dar. In Kombination mit den gewählten AS-Nummern, deren IP-Bereich und den IP-Adressen der Schnittstellen am jeweiligen Router ließe sich so ein vollvermaschtes Netzwerk der AS herstellen. Jedes AS kommuniziert somit mit jedem der anderen Teilnehmer am simulierten Internet-Knoten.

4.3. Hardware

Dieser Abschnitt beschreibt die gewählte Hardware, die eingesetzt wird, um die in Abschnitt 4.2 beschriebene Topologie des Versuchsaufbaus umzusetzen.

An einem Internet-Knoten schließen sich eine Vielzahl von AS und damit einhergehend eine Vielzahl von Geräten, oftmals von verschiedenen Herstellern und verschiedenen Architekturen, zusammen. Um ein reales Bild eines Internet-Knotens darzustellen, wird ein heterogenes Netzwerk aus verschiedenen Geräten gewählt, bei denen sich nicht nur die Modelle, sondern auch die Hersteller der Geräte unterscheiden. Zur Veranschaulichung dieser Heterogenität werden verschiedene Hardware-Anforderungen bei der Wahl der Geräte berücksichtigt. Die Wahl fiel auf drei Raspberry Pi sowie einen Orange Pi. Ein Raspberry Pi Model 4 B, zwei Raspberry Pi Model 3 B+ und einen Orange Pi Zero Plus.
Für den zentralen Switch, der die Einplatinencomputer verbindet, wird ein Switch mit 5 Ethernet-Ports, mit einer Kapazität pro Port von jeweils einem Gigabit pro Sekunde verwendet. Ein Ethernet Switch von TP Link, Modell TL-SG-105 zu sehen in Abbildung 4.3, bietet hierbei die grundlegenden Funktionen.

Abbildung 4.3.: TP-Link SG105
Quelle: https://www.tp-link.com/de/business-networking/unmanaged-switch/tl-sg105/

Die Geräte werden per Ethernet-Kabel an den Switch angeschlossen. Alle fünf verwendeten Geräte, die vier Einplatinencomputer und der Ethernet Switch, benötigen jeweils ein externes Netzteil zur Stromversorgung. Weiterhin benötigen alle Einplatinencomputer eine SD-Karte zum Bespielen eines Betriebssystems und zum Installieren von Software. Zusätzlich werden Peripherie-Geräte, sprich eine Tastatur, eine Maus und ein Monitor mit HDMI Anschluss sowie ein HDMI-Kabel, zur Steuerung und zur erstmaligen Einrichtung der Raspberry Pi Geräte, benötigt.
Die Stromversorgung wird über eine Steckdosenleiste mit 2-poligen Schalter an jeweils einen Netzstecker für jedes Gerät geliefert. Der Schalter dient zur simultanen und kontrollierten Ab- und Einschaltung aller Geräte im Versuchsaufbau, da die filigranen Einplatinencomputern empfindlich auf Spannung reagieren können.

4.4. Versuchsablauf

Dieser Abschnitt stellt den Versuchsablauf zur Simulation eines Internet-Knotens dar. Es werden die benötigten Arbeitsschritte aufgelistet und eingesetzte Komponenten beleuchtet.

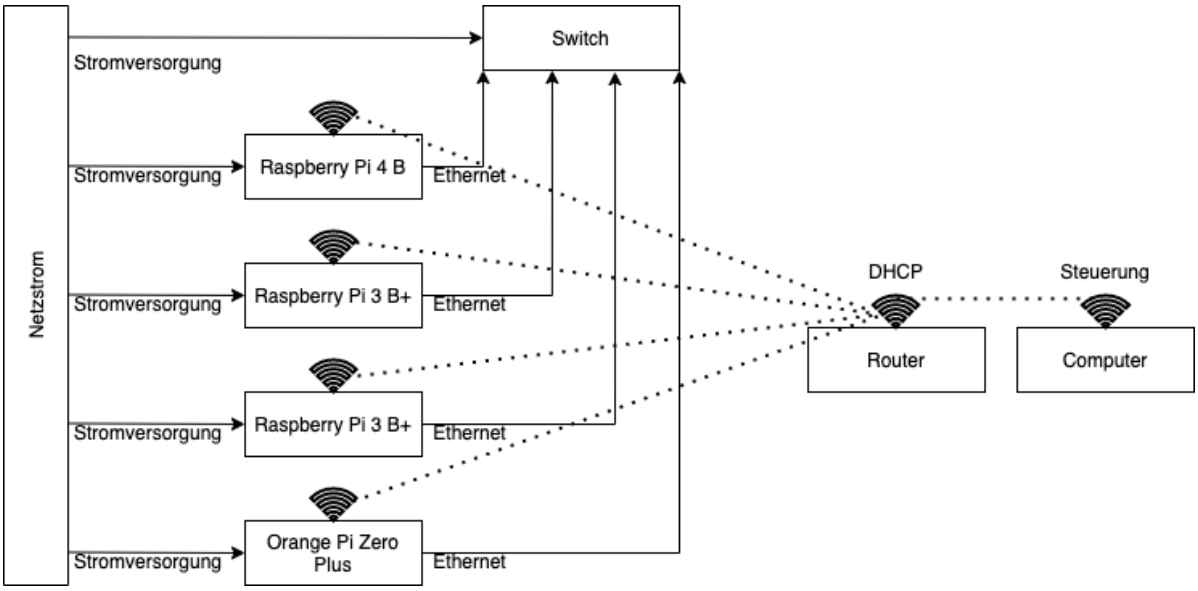

Abbildung 4.4.: Versuchsaufbau Komponenten

Abbildung 4.4 stellt die eingesetzten Komponenten im Versuchsaufbau dar. Links im Bild wird schematisch die Stromversorgung angedeutet. Alle gewählten Einplatinencomputer bieten jeweils eine Ethernet-Schnittstelle und eine WLAN-Schnittstelle. Die Ethernet-Schnittstelle dient als Verbindung zu den anderen Geräten im Versuchsaufbau, sie wird zur Inter-AS-Kommunikation verwendet. Die WLAN-Schnittstelle dient zur Bedienung, als sogenannter „Management-Link“, also der Verwaltung per Fernsteuerung. Die Vergabe der WLAN IP-Adressen erfolgt per Dynamic Host Configuration Protocol (DHCP) über einen zusätzlichen Router. Dabei hat dieser Router keinen Einfluss auf das Geschehen während der Kommunikation zwischen den AS und dient lediglich zum Verbindungsaufbau mit dem Internet während der Installation und zur Fernsteuerung über WLAN. Alle Einplatinencomputer bieten zudem die Möglichkeit der Steuerung per Kommandozeile, beziehungsweise per Secure Shell (SSH), welche im Verlauf der Implementierung und des Versuchs über die WLAN-Schnittstelle eingesetzt wird. Dies bietet einen Vorteil gegenüber der Bedienung jedes einzelnen Gerätes über Tastatur, Maus und Monitor. So können von einem Computer über eine Kommandozeile alle Geräte bedient werden.
Als Betriebssystem für alle verbundenen Geräte wird eine Linux Distribution gewählt. Linux Distributionen bieten den Vorteil einer quelloffenen und kostenlosen Umgebung, in der eine Vielzahl der benötigten Software ebenfalls quelloffen und kostenlos frei verfügbar ist. Für die Raspberry Pi Geräte wird das Betriebssystem der Raspberry Pi Foundation, „Raspbian“ beziehungsweise „Raspberry Pi OS“, verwendet. Für den Orange Pi Einplatinencomputer wird die Linux Distribution „Armbian“ verwendet. Beide Distributionen basieren auf der ebenfalls quelloffenen und kostenlosen Linux Distribution „Debian“.
Es werden statische IP-Adressen (siehe Tabelle 4.1) auf den Ethernet-Schnittstellen der jeweiligen Geräte eingerichtet, um einen Netzwerkbereich eines AS zu simulieren. Bei der Inter-AS-Kommunikation kommt FRRouting zum Einsatz, bei der die Software-Konfiguration die Netzwerk Topologie abbildet. Abbildung 4.5 zeigt den aufgebauten Versuchsaufbau mit allen verbauten Komponenten.

Abbildung 4.5.: Versuchsaufbau

5. Implementierung

Dieses Kapitel behandelt die Implementierung des Aufbaus. Es werden die Schritte zur Konfiguration der Einplatinencomputer sowie zur Konfiguration der Schnittstellen dargestellt, um das in Kapitel vier beschriebene Design umzusetzen. Weiterhin wird eine Netzwerkkonfiguration auf allen Geräten vorgenommen, um die Kommunikation zwischen allen Geräten zu ermöglichen, eine Konfiguration für FRRouting gewählt, um eine AS übergreifende Kommunikation aufzubauen sowie Kommandozeilenwerkzeuge und Software verwendet, um die implementierten Funktionen zu verifizieren.

5.1. Einrichtung von Raspberry Pi OS

Der folgende Abschnitt beschreibt das Erstellen eines Betriebssystemabbildes für die Raspberry Pi Plattform. Die hier genannten Arbeitsschritte sind für alle drei Raspberry Pi Geräte durchgeführt worden.

Es erfolgt zunächst die Installation des Raspberry Pi OS Betriebssystems (Raspbian GNU/Linux 10, Linux Kernel 5.10.63-v7+) auf allen Raspberry Pi Geräten. Das Betriebssystem ist zu allen Raspberry Pi Modellen kompatibel und kann mit dem dazugehörigen Programm „Raspberry Pi Imager“ auf eine zuvor formatierte SD-Karte aufgespielt werden . Das Aufspielen des Betriebssystems auf die SD-Karte erfolgt durch das Programm „Raspberry Pi Imager“ (Version 1.6.2) mittels eines Knopfdrucks, wie in Abbildung 5.1 dargestellt. Das erfolgreiche Beschreiben des Betriebssystems wird mit einer Meldung auf dem Bildschirm bestätigt.

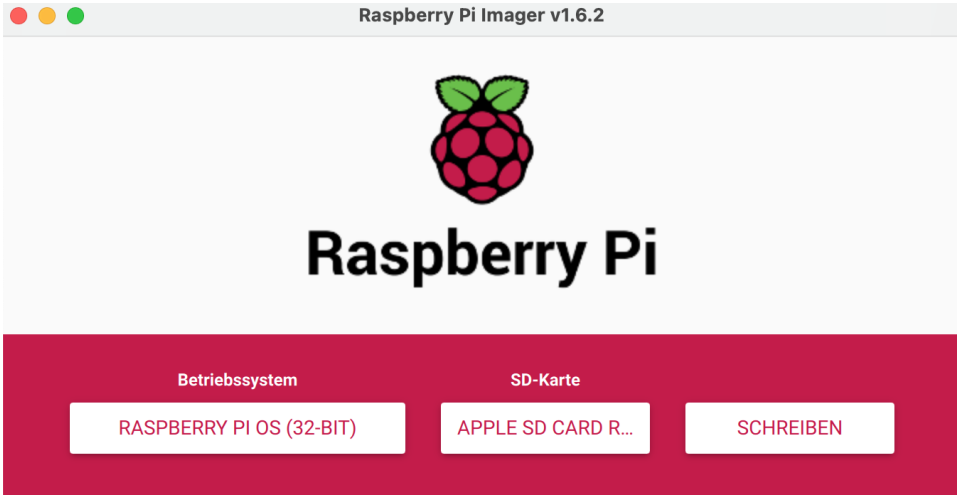

Abbildung 5.1.: Raspberry Pi Imager

Ist das Betriebssystem auf der SD-Karte aufgespielt, kann der Raspberry Pi bereits in Betrieb genommen werden. Neben der notwendigen Stromversorgung wird für das erstmalige Starten ein Monitor über die HDMI-Schnittstelle des Raspberry Pi sowie eine Maus und Tastatur an die USB-Schnittstelle angeschlossen. Beim erstmaligen Starten wird die grafische Benutzeroberfläche gestartet, zu sehen in Abbildung 5.2. Es folgt ein Menü für Systemeinstellungen, unter anderem für die Systemsprache, Tastaturbelegung, zur Änderung des Standardpassworts des Raspberry Pi und der Verbindung mit einem Netzwerk.

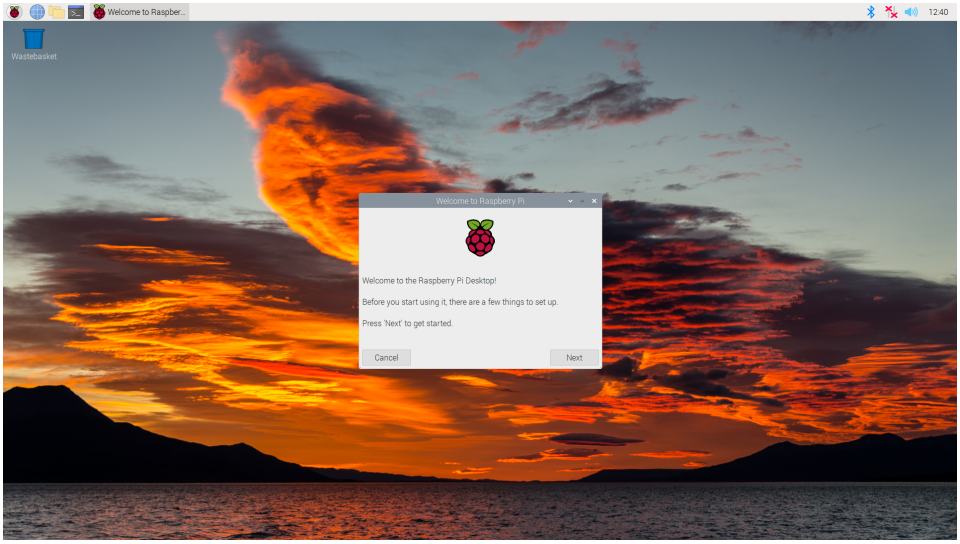

Abbildung 5.2.: Raspberry Pi OS grafische Oberfläche

Für den Versuchsaufbau wird für die einfachere Handhabung mit allen Geräten der SSH-Dienst des Raspberry Pi aktiviert. Dieser ist bereits vorab installiert und kann mit einem einfachen Knopfdruck gestartet werden, wie in Abbildung 5.3 zu sehen ist.

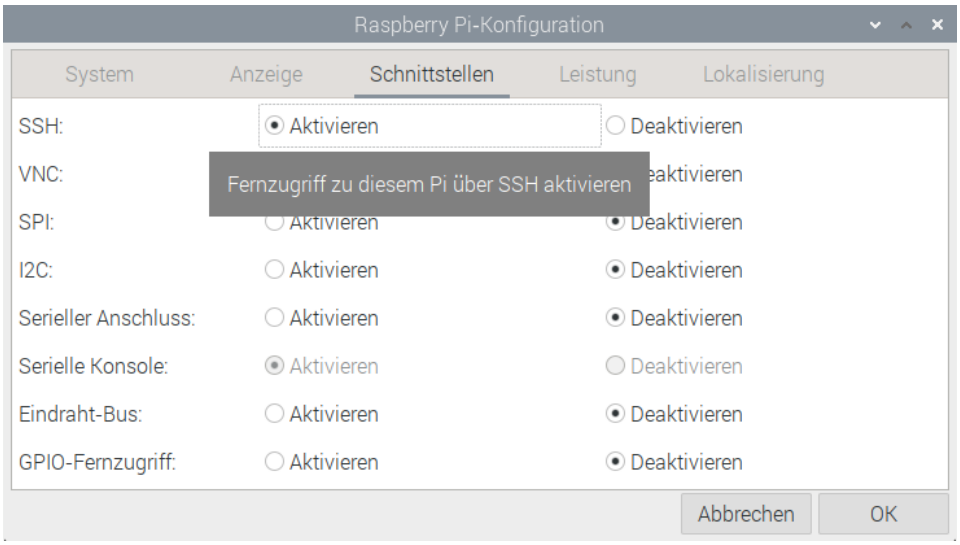

Abbildung 5.3.: Raspberry Pi OS-SSH Dienst Aktivierung

Der erstmalige Startvorgang des Raspberry Pi endet mit dem Aktualisieren der Pakete sowie einem abschließenden Neustart des Systems. Nach dem Neustart wird das Gerät per SSH Fernzugriff und der Kommandozeile bedient.

```
$ ssh pi@192.168.0.160
pi@192.168.0.160's password:
Linux raspberrypi 5.10.63-v7+ #1459 SMP Wed Oct 6 16:41:10 BST 2021 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Dec 18 22:24:58 2021 from 192.168.0.58
pi@raspberrypi:~ $
```

Abbildung 5.4.: Raspberry Pi erstmalige SSH Verbindung

Abbildung 5.4 zeigt die Ausgabe der erstmaligen SSH Verbindung. Die Raspberry Pi Geräte werden nach der Konfiguration an den Ethernet Switch angeschlossen.

5.2. Einrichtung von Armbian

Der Orange Pi Zero Plus benötigt ein anderes Betriebssystem als die Raspberry Pi Geräte, da das Raspberry Pi OS zu der Architektur des Orange Pi nicht kompatibel ist. Als Alternative wird daher das auf der Debian Distribution basierende, Betriebssystem Armbian (Armbian 21.08.3 Focal, Linux Kernel 5.10.60-sunxi64) verwendet.

Die Entwickler von Armbian geben auf ihrer Webseite einige Instruktionen zur Installation an [4]. Wie in Abbildung 5.5 dargestellt, wird die vorformatierte SD-Karte über das Programm „balenaEtcher“ bespielt. Das Aufspielen des Betriebssystems auf eine SD-Karte erfolgt ähnlich zum Raspberry Pi OS über einen Knopfdruck.

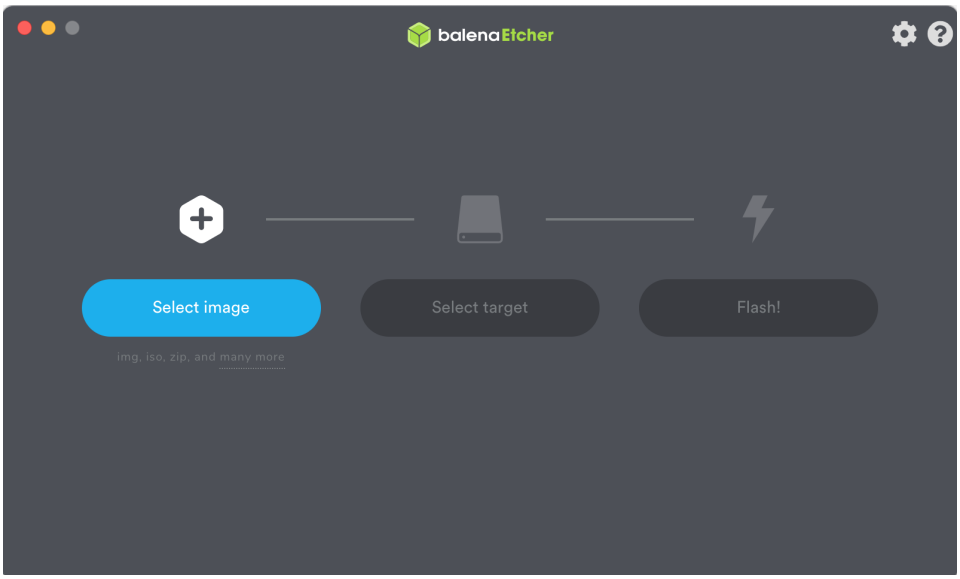

Abbildung 5.5.: balenaEtcher

Im Gegensatz zum Raspberry Pi OS wird beim erstmaligen Starten von Armbian die Verbindung über SSH aufgebaut, da der Orange Pi Zero Plus keine HDMI Schnittstelle besitzt. Für die erstmalige Verbindung wird die Ethernet-Schnittstelle verwendet. Die Entwickler geben hierzu das Standard-Passwort (siehe Tabelle 5.1) auf ihrer Webseite an.

Tabelle 5.1.: Standard Passwort Armbian

Benutzer	root
Passwort	1234

Nach der Anmeldung folgt zunächst die Aufforderung das Passwort zu ändern. Danach folgt die Verbindung an den Router über die WLAN-Schnittstelle und die Aktualisierung des Systems sowie ein Neustart. Nach der Verbindung mit dem Router kann über die WLAN-Schnittstelle mit dem Gerät kommuniziert werden. Das Gerät wird nun in den Verbund angeschlossen beziehungsweise an den Ethernet Switch verbunden.

5.3. Konfiguration der Schnittstellen

Dieser Abschnitt stellt die Schritte dar, die benötigt werden, um die Schnittstellen für die Geräte im Versuchsaufbau zu konfigurieren. Es folgen die Konfiguration der WLAN und Ethernet-Schnittstellen. Über den lokalen Router wird sichergestellt, dass die WLAN-Schnittstelle per DHCP (Dynamic Host Configuration Protocol) eine IP-Adresse erhält. Dies stellt die Schnittstelle zum Fernzugriff sicher. Die Ethernet-Schnittstelle wird so konfiguriert, dass das später eingesetzte FRRouting im BGP-Router einen eigenen Adressbereich erhält. Da sich die Konfigurationen der beiden Betriebssysteme unterscheiden folgen nun zwei Abschnitte jeweils für das Raspberry Pi OS und Armbian.

5.3.1. Konfiguration unter Raspberry Pi OS

Die Zuweisung des IP-Adressbereichs erfolgt gemäß der Tabelle 4.1 für jedes Gerät. Um dies zu erzielen wird eine statische IP-Adresse benötigt, welche sich zur Laufzeit nicht ändert. Sie emuliert die für ein AS typischen IP-Adressbereich. Beim Raspberry Pi OS wird dies über den DHCPCD-Dienst erzielt. Dabei muss man einen Eintrag in der Dienst-Konfigurationsdatei erstellen. Diese Datei befindet sich in der Verzeichnisstruktur des Systems unter `/etc/dhcpcd.conf`.

Dem Anhang A ist die vollständige Datei zu entnehmen. Darin enthalten sind die Konfigurationen der Schnittstellen für alle Raspberry Pi Geräte. Exemplarisch für alle drei Geräte wird nun auf die Konfiguration für AS10 in Zeile 70 bis 73 Anhang A eingegangen.

```
70  interface eth0
71  static ip_address=10.0.1.1/24
72  static routers=10.0.1.1
73  static domain_name_servers=10.0.1.1
```

Mit dem Eintrag in Zeile 70 weist man der Schnittstelle `eth0`, in diesem Fall der Ethernet-Schnittstelle des Raspberry Pi, eine statische IP-Adresse zu (Zeile 71). Zudem werden der Standard-Router und der Domain-Namens-Server auf die eigene IP-Adresse (Zeile 72 und Zeile 73) gesetzt. Sodass das, nun simulierte, AS seinen eigenen IP-Adressbereich besitzt und nicht mit anderen Systemen kommuniziert. Ein Neustart des DHCPCD-Dienstes macht die Änderungen wirksam. Mittels `systemctl` wird der Dienst automatisch beim Hochfahren des Gerätes gestartet:

```
systemctl enable dhcpcd
```

Da der Raspberry Pi die Ethernet-Schnittstelle der WLAN-Schnittstelle bevorzugt, muss sichergestellt werden, dass bei verbundenem Ethernet-Kabel weiterhin der SSH-Dienst über die WLAN-Schnittstelle angesprochen werden kann. Das Gerät könnte sich sozusagen „aussperren“, da die konfigurierte IP-Adresse für die Ethernet-Schnittstelle nicht im Adressbereich (`192.168.0.0/24`) des DHCP-Routers liegt. Dafür wird eine statische Route zum Router eingerichtet. Dies erfolgt mittels des ip Kommandos, welches eine Route zum Netzwerk des DHCP-Routers über die IP-Adresse `192.168.0.1` konfiguriert. Dabei wird die Schnittstelle `wlan0` (WLAN-Schnittstelle) verwendet:

```
ip route add 192.168.0.0/24 via 192.168.0.1 dev wlan0
```

Statische Routen sind allerdings nicht permanent und werden beim Neustart des Gerätes gelöscht. Um eine persistente Route zu erhalten, die nach dem Neustart des Gerätes nicht wieder gelöscht wird, muss eine Konfigurationsdatei im Verzeichnis des DHCPCD-Dienstes (`/lib/dhcpcd/dhcpcd-hooks/`) erstellt werden. Diese Datei wird beim Starten des Dienstes gelesen und angewendet:

```
/lib/dhcpcd/dhcpcd-hooks/40-route:
ip route add 192.168.0.0/24 via 192.168.0.1 dev wlan0
```

Die Namensgebung der Datei ist hierbei nebensächlich. Jedoch stellt die Zahl die Reihenfolge dar, in welcher die Dateien eingelesen werden. Hierbei wurde die Zahl 40 festgelegt, da zuvor wichtige Konfigurationsdateien eingelesen werden. Damit kann das Gerät trotz verbundenem Ethernet-Kabel und aktiver Ethernet-Schnittstelle korrekt über die WLAN-Schnittstelle kommunizieren.

Für die Netzwerk-Topologie aus Kapitel vier werden weitere IP-Adressen und Schnittstellen benötigt. Da die Einplatinencomputer jeweils nur eine physische Schnittstelle

(eth0) besitzen, werden sekundäre (und tertiäre) IP-Adressen für die Schnittstellen konfiguriert. Das Konfigurieren solcher zusätzlichen IP-Adressen erfolgt mit dem ip Kommando. Der folgende Befehl weist der Schnittstelle eth0 eine weitere IP-Adresse zu. Durch die Doppelpunkt-Notation „eth0:0“ wird der Schnittstelle eth0 eine weitere IP-Adresse zugewiesen:

```
ip address add 10.0.1.20/24 dev eth0:0
```

Um diese Zuweisung persistent zu halten, sollte bei jedem Neustart des Systems dieser Befehl ausgeführt werden. Dies wird erzielt, indem dieser Befehl in die Datei /etc/rc.local hinzugefügt wird. Einträge in dieser Datei werden beim Systemstart ausgeführt. Anhang C zeigt die Einträge für jedes Gerät im Versuchsaufbau. Die Zeilen 14 bis 16 stellen die Einträge für das AS10 dar. Die Zeilen 18 bis 20 die des AS20, die Zeilen 22 und 23 für das AS30 und die Zeilen 25 und 26 stellen die Einträge für das AS40 dar.

5.3.2. Konfiguration unter Armbian

Bei der Schnittstellen-Konfiguration unter Armbian wird ein Eintrag in die Datei /etc/network/interfaces hinzugefügt. Anhang B zeigt hierbei die gesamte Datei. In Zeile 11 bis 15 wird die Schnittstelle eth0 konfiguriert, der Schnittstelle eine statische IP-Adresse (10.0.3.1) zugewiesen sowie die Netzwerkmaske (255.255.255.0 = /24) und den Standard-Router (gateway) auf die eigene IP-Adresse gesetzt. Die Zeilen 17 und 18 dienen als alternative für die WLAN-Schnittstelle (wlan0). Die Zuweisung der IP-Adresse erfolgt durch den DHCP Router automatisch. Ein Neustart des Netzwerk-Dienstes mit systemctl restart networking bewirkt die Änderung in der interfaces Datei.

Für die dauerhafte Verfügbarkeit der WLAN-Schnittstelle bei angeschlossenem Ethernet-Kabel wird hierbei ebenfalls eine statische Route zum DHCP Server konfiguriert (siehe vorheriger Abschnitt). Unter Armbian bleibt diese Route persistent, der Eintrag in der Routing-Tabelle des Systems bleibt nach einem Neustart erhalten.

Die Konfiguration einer sekundären IP-Adresse für die Ethernet-Schnittstelle ist bei dem Orange Pi beziehungsweise dem Armbian Gerät ebenfalls von Nöten. Die Arbeitsschritte sind analog zu denen des Raspberry Pi OS. Die Einträge der Schnittstellen werden ebenfalls in der Datei /etc/rc.local hinzugefügt (siehe Anhang C).

5.4. FRRouting Installation

Als nächstes wird FRRouting (Version 8.0.1) auf allen Geräten installiert. Dafür bieten die Entwickler der Software mehrere Möglichkeiten[1], zum Beispiel der manuellen Installation, indem die Installationsdateien manuell heruntergeladen, konfiguriert und ausgeführt werden oder der Installation über verschiedene sogenannte Paketmanager. Hierbei wurde sich für die Installation über den Debian Paketmanager entschieden, da eine genaue Konfiguration der Installation nicht notwendig ist und die Betriebssysteme Raspberry Pi OS und Armbian im Kern auf Debian basieren. Die Betreiber des Paketmanagers listen die nötigen Arbeitsschritte zum Installieren auf ihrer Webseite auf[2].
In Anhang D sind die durchgeführten Installationsschritte für FRRouting aufgelistet. Zur Authentifizierung des Paketes müssen die GPG (GNU Privacy Guard) Schlüssel in den lokalen Schlüsselbund hinzugefügt werden. Danach wird die Softwareversion auf die möglichst aktuelle gesetzt und anschließend über den Paketmanager `apt` heruntergeladen. Die Installation erfolgt automatisch. Hierbei wird ein Benutzer `frr` angelegt sowie weitere benötigte Dienste wie beispielsweise eine interaktive Kommandozeile installiert, welche mit `vtysh` gestartet wird [16].

Unter dem Verzeichnis `/etc/frr/` finden sich nach der Installation die Konfigurationsdateien für FRRouting. Zur Aktivierung des benötigten BGP-Dienstes wird in der Datei `deamons` der Eintrag `bgpd=no` auf `bgpd=yes` gesetzt. Anhang E listet die vollständige Datei auf.

Weiterhin fordert die Installation ein Bearbeiten der Datei `/etc/services` [16]. Hierbei müssen die Dienste, die durch FRRouting implementiert werden, einem Port zugewiesen werden. Dies erfolgt durch Hinzufügen einiger Zeilen in die Datei. Die benötigten Einträge sind in Anhang F aufgelistet. Dies umfasst bereits alle benötigten Arbeitsschritte, bevor man FRRouting erstmalig starten kann. Um FRRouting zu starten, sendet man einen `start` Befehl an die bereits installierte Systemdatei unter `/etc/init.d/frr`:
`/etc/init.d/frr start`

[1] https://docs.frrouting.org/en/latest/installation.html
[2] https://deb.frrouting.org

5.5. FRRouting Konfiguration

FRRouting ist ohne Konfiguration inaktiv. Daher ist der nächste Schritt eine passende BGP Konfiguration zu erstellen, welche die Netzwerk-Topologie abbildet und eine Inter-AS-Kommunikation ermöglicht. Alle Konfigurationen der AS sind unter den Anhängen G bis J in Gänze zu finden. Exemplarisch wird im Folgenden auf die Konfiguration für AS10 vertiefend eingegangen, zu finden in Anhang G. Die anderen Konfigurationen erfolgen ähnlich, jedoch mit unterschiedlichen IP-Adressen.

Die Zeilen 1 bis 11 in Anhang G stellen automatisch generierte Zeilen des Programms dar. Zeilen mit einem beginnenden Ausrufezeichen stellen Kommentarzeilen dar und werden vom Programm ignoriert. Es folgen Auszüge aus Anhang G, auf die nun vertiefend eingegangen wird.

```
12  router bgp 10
13  bgp router-id 10.0.1.1
14  bgp bestpath as-path multipath-relax
15  neighbor peer peer-group
16  # AS 20
17  neighbor 10.0.1.20 remote-as 20
18  neighbor 10.0.1.20 peer-group peer
19  # AS 30
20  neighbor 10.0.3.1 remote-as 30
21  neighbor 10.0.3.1 peer-group peer
22  # AS 40
23  neighbor 10.0.4.1 remote-as 40
24  neighbor 10.0.4.1 peer-group peer
```

Mit Zeile 12 beginnt die Konfiguration des AS10. Es wird dem Router die Funktion für BGP sowie die eigene AS-Nummer 10 zugewiesen. In Zeile 13 wird dem Router die eigene IP-Adresse `10.0.1.1` zugewiesen. Anschließend folgt in Zeile 14 die Ermittlung des besten Pfades für BGP. Die Konfiguration `multipath-relax` ermöglicht die Betrachtung aller Pfade bei mehreren Quellen. Zeile 15 fügt eine Gruppe `peer` hinzu. Diese ermöglicht es, Nachbar-AS in Gruppen zusammenzufassen und entsprechende Einstellungen für alle AS in dieser Gruppe vorzunehmen. Die Zeilen 16 bis 24 fügen alle Peering-Partner, also Nachbar AS, hinzu. Dabei wird die IP-Adresse der Nachbar-AS angegeben und diese in die Gruppe `peer` hinzugefügt.

```
26  network 10.0.1.0/24
27  neighbor peer route-map peering-in in
28  neighbor peer route-map peering-out out
29  neighbor peer activate
30  exit-address-family
31  !
32  route-map upstream-out permit 100
33  set local-preference 10
34  !
35  route-map upstream-in permit 100
36  set local-preference 10
37  !
38  route-map peering-in permit 100
39  set local-preference 1000
40  !
41  route-map peering-out permit 100
42  set local-preference 1000
43  !
```

In Zeile 26 bis 31 wird dem Router der IPv4-Adressbereich `10.0.1.0/24` zugewiesen, alle Nachbar Netzwerke der Gruppe „peer“ sowie eine Route-Map, ein Entscheidungs-Regelwerk für einkommende und ausgehende Pakete, zugewiesen. Anschließend wird diese Zuweisung aktiviert.
Die restlichen Zeilen 33 bis 43 erstellen die zuvor zugewiesenen Routing-Entscheidungsregeln. Bei einkommenden Pakten wird in der `peering-in` und `peering-out route-map` die `local-preference` auf den Wert 1000 gesetzt. Diese wird bei Peering Verbindungen angewendet. Bei einer möglichen Transit Verbindung könnte die `route-map upstream-in` und `upstream-out` verwendet werden, um einen `local-preference` Wert von 10 zu konfigurieren. Das Schlüsselwort `permit` gibt an, dass bei der jeweiligen `route-map` die Weiterleitung der BGP-Nachrichten erlaubt wird, da im Versuchsaufbau eine vollvermaschte Topologie umgesetzt wird.
Zur Aktivierung der erstellten Konfigurationen muss der FRRouting Dienst neu gestartet werden. Dies erfolgt durch das entsprechende systemctl Kommando:
`systemctl restart frr.service`

5.6. Verifizierung der Implementierung

In diesem Abschnitt folgt die Verifizierung der in diesem Kapitel durchgeführte Implementierung mit Hilfe der in Kapitel 3.5 vorgestellten Kommandozeilenwerkzeuge. Zuvor werden die im vorherigen Abschnitt erläuterte FRRouting Konfiguration im Programm selbst verifiziert. Die Ausgaben der jeweiligen Kommandos sind anhand von Bildschirmfotos dargestellt.

Um mit dem BGP Router zu interagieren, wird die von FRRouting verwendete interaktive Eingabefenster `vtysh` verwendet. Nach dem Aufruf des Kommandos mit Administratorrechten erfolgt eine Begrüßungsnachricht, wie in Abbildung 5.6 zu sehen. Ebenfalls ändert sich das zuvor verwendete Eingabefenster.

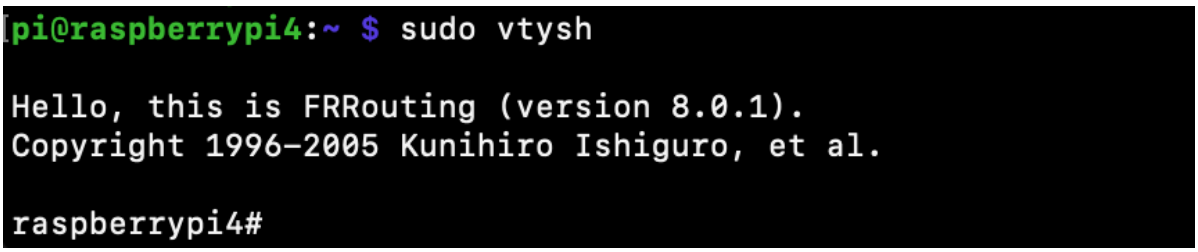

Abbildung 5.6.: FRRouting Login

`vtysh` ermöglicht dem Benutzer das Navigieren und interaktive Konfigurieren des durch FRRouting implementieren Routers.
Befehle wie `show bgp neighbor` geben detaillierte Informationen über die aktuell verbundenen Nachbar-AS aus. Der Befehl `show bgp summary` gibt eine Zusammenfassung des `show bgp neighbor` Befehls aus und wird daher im Folgenden näher beleuchtet.

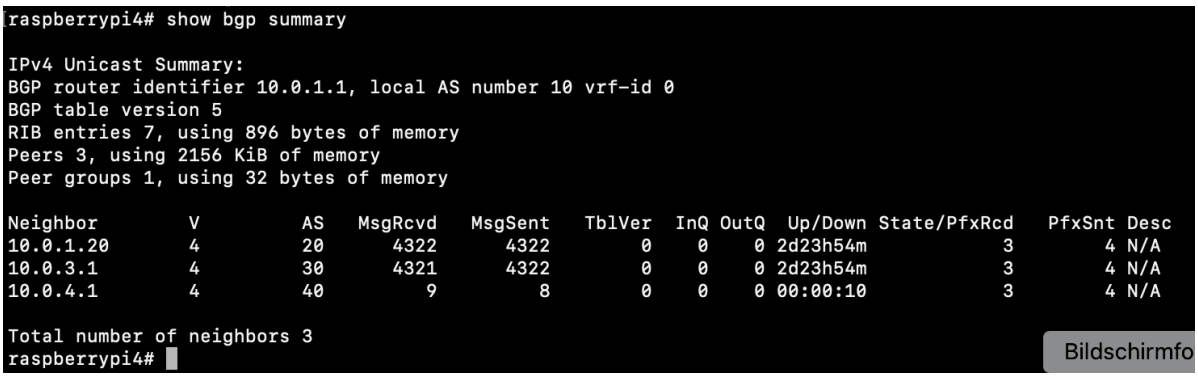

Abbildung 5.7.: AS10 BGP Zusammenfassung

Abbildung 5.7 stellt die Ausgabe für AS10 dar. Weitere Informationen wie die IP-Adresse des BGP-Routers von AS10 sowie die AS-Nummer werden ebenfalls ausgegeben. Die Anzahl der Peering-Verbindungen („peers“) beträgt drei. Die IP-Adressen der Router der zuvor konfigurierten Nachbar-AS sind in der Spalte „Neighbor“ zu finden. Der Wert vier in der Spalte `V` indiziert das verwendete IP-Protokoll Version 4 (IPv4). Die AS-Nummern der Nachbar-AS sind in der Spalte `AS` angezeigt. Die Spalten `MsgRcvd` und `MsgSent` listen die Anzahl an erhaltenen und gesendeten BGP-Nachrichten (Keepalive-Nachrichten) während der BGP-Sitzung. Zu Demonstrationszwecken wurde die Verbindung zu AS40 neugestartet. In der Spalte „Up/Down“ wird die Dauer der etablierten BGP-Sitzung angezeigt. Wie zu sehen ist, ist die Verbindung zu AS40 seit 10 Sekunden aktiv. Dabei wurden neun Nachrichten empfangen und acht Nachrichten versendet. Die Verbindungen zu AS20 und AS30 sind in der Abbildung bereits seit 2 Tagen 23 Stunden und 54 Minuten aktiv. Dabei wurden jeweils über 4000 Nachrichten empfangen und gesendet. Die Spalten `PfxRcd` und `PfxSnt` geben die Anzahl an empfangenen und gesendeten Präfixen von den jeweiligen BGP-Verbindungen an.
Weiterhin sieht man den für die Routing-Informationen verbrauchten Arbeitsspeicherplatz. In diesem Fall werden für sieben Routen-Einträge (`RIB entries`) 896 Byte Speicher, für die Information von drei Peering-Partner (`peers`) 2156 Kilobyte Speicher und für die Information über die Peering-Gruppe (`Peer Groups`) werden 32 Byte Speicher verbraucht.

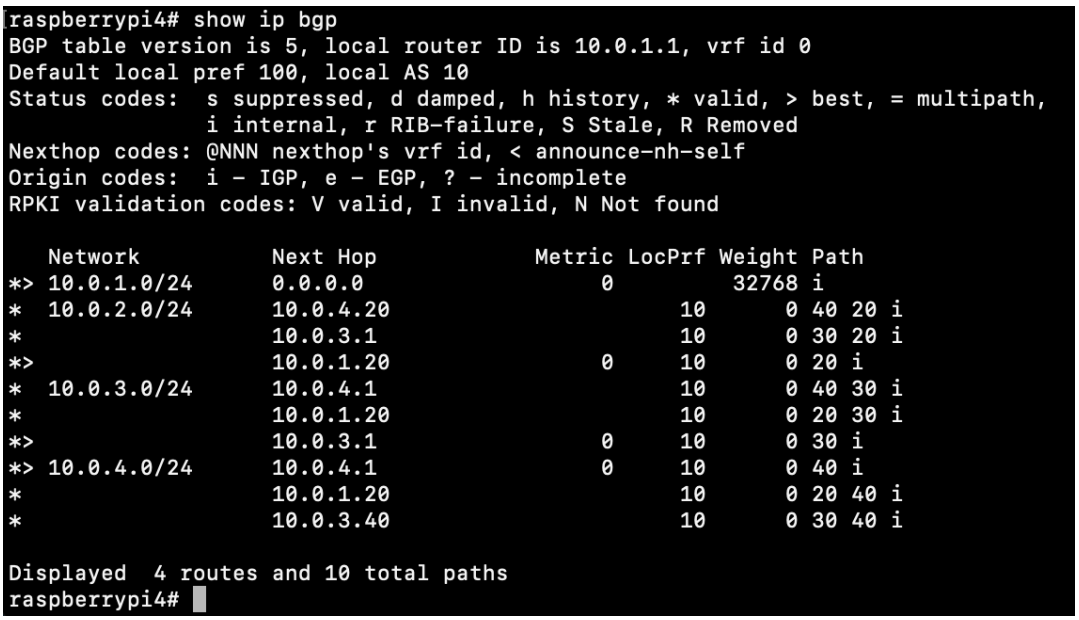

Abbildung 5.8.: AS10 IP BGP Verbindungen

Weitere Befehle zur Ausgabe von bestehenden Verbindungen sind mit `show ip bgp` und `show ip route` verfügbar. Die Ausgabe des Befehls `show ip bgp` ist in Abbildung 5.8 dargestellt. Zu sehen sind die Erklärungen der jeweiligen Abkürzungen des Routers. Die Spalte `Network` gibt den IP-Adressbereich des jeweiligen Nachbar-AS an. In der Spalte `Next Hop` ist die IP-Adresse des Netzwerk-Routers angegeben, welche im AS-Pfad an nächster Stelle steht. Die Spalten `Metric`, `LocPrf` und `Weight` geben BGP spezifische Werte zur Ermittlung des besten Pfades an. Der gesamte AS-Pfad ist in der Spalte `Path` abgebildet. Der ermittelte beste Pfad wird links im Bild mit einem Pfeil „>“ markiert.

```
raspberrypi4# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

K * 0.0.0.0/0 [0/303] via 192.168.0.1, wlan0, src 192.168.0.206, 05:12:01
K>* 0.0.0.0/0 [0/202] via 10.0.1.1, eth0, src 10.0.1.1, 3d00h38m
K * 10.0.1.0/24 [0/202] is directly connected, eth0, 3d00h38m
C>* 10.0.1.0/24 is directly connected, eth0, 3d00h38m
B>* 10.0.2.0/24 [20/0] via 10.0.1.20, eth0, weight 1, 3d00h38m
B   10.0.3.0/24 [20/0] via 10.0.3.1 inactive, weight 1, 3d00h38m
C>* 10.0.3.0/24 is directly connected, eth0, 3d00h39m
B   10.0.4.0/24 [20/0] via 10.0.4.1 inactive, weight 1, 00:44:51
C>* 10.0.4.0/24 is directly connected, eth0, 3d00h39m
K * 192.168.0.0/24 [0/0] via 192.168.0.1, wlan0 inactive, 05:12:01
K * 192.168.0.0/24 [0/303] is directly connected, wlan0, 05:12:01
C>* 192.168.0.0/24 is directly connected, wlan0, 05:12:01
raspberrypi4#
```

Abbildung 5.9.: AS10 IP Routen

```
raspberrypi# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

K * 0.0.0.0/0 [0/303] via 192.168.0.1, wlan0, src 192.168.0.160, 01:33:08
K>* 0.0.0.0/0 [0/202] via 10.0.4.1, eth0, src 10.0.4.1, 1d13h28m
B>* 10.0.1.0/24 [20/0] via 10.0.4.10, eth0, weight 1, 1d13h28m
B>* 10.0.2.0/24 [20/0] via 10.0.4.20, eth0, weight 1, 1d13h28m
B   10.0.3.0/24 [20/0] via 10.0.3.1 inactive, weight 1, 1d13h28m
C>* 10.0.3.0/24 is directly connected, eth0, 1d13h28m
K * 10.0.4.0/24 [0/202] is directly connected, eth0, 1d13h28m
C>* 10.0.4.0/24 is directly connected, eth0, 1d13h28m
K>* 169.254.0.0/16 [0/303] is directly connected, wlan0, 3d20h38m
K * 192.168.0.0/24 [0/0] via 192.168.0.1, wlan0, 01:33:08
K * 192.168.0.0/24 [0/303] is directly connected, wlan0, 01:33:08
C>* 192.168.0.0/24 is directly connected, wlan0, 01:33:08
raspberrypi#
```

Abbildung 5.10.: AS40 IP Routen

Zur Darstellung aller Routen wird der Befehl `show ip route` verwendet. Die Ausgabe des Befehls ist in Abbildung 5.9 für AS10 und in 5.10 für AS40 dargestellt. Ebenfalls werden alle Abkürzungen der Ausgabe vom Programm erläutert. Die in der Abbildung dargestellten Einträge stellen die Routen dar, die im System verfügbar sind. Alle Routen, die links mit einem „K“ markiert sind, sind Routen, die vom Kernel implementiert werden. Routen denen ein „C“ vorausgeht, sind direkt verbundene Schnittstellen beziehungsweise Netzwerke. Die mit „B“ markierten Routen stellen Routen dar, die über

BGP dynamisch hinzugefügt wurden. Die ausgewählten Routen werden links im Bild mit einem Pfeil „>“ markiert.
Ein Eintrag einer Route besteht aus dem Präfix des Netzwerks, zu dem die Route führt, also zum Ziel-Netzwerk. Es wird die Gewichtung der Route in den eckigen Klammern angezeigt sowie die IP-Adresse des Routers, der für diese Route als nächstes Ziel konfiguriert ist. Der Name der Schnittstelle, über den die Route geleitet wird, sowie die vergangene Zeit, seitdem diese Route aktiv ist, werden ebenfalls ausgegeben.
Beispielsweise die Route zum Netzwerk `10.0.2.0/24` ist eine über BGP dynamisch hinzugefügte Route, die über die Schnittstelle `eth0` und die IP-Adresse `10.0.1.20` geleitet wird. Sie wurde als beste Route ausgewählt und ist seit drei Tagen und 38 Minuten aktiv. Als Vergleich ist in Abbildung 5.10 die Ausgabe für AS40 dargestellt.

```
pi@raspberrypi4:~ $ ping 10.0.2.1 -c 3
PING 10.0.2.1 (10.0.2.1) 56(84) bytes of data.
64 bytes from 10.0.2.1: icmp_seq=1 ttl=64 time=0.342 ms
64 bytes from 10.0.2.1: icmp_seq=2 ttl=64 time=0.488 ms
64 bytes from 10.0.2.1: icmp_seq=3 ttl=64 time=0.339 ms

--- 10.0.2.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 100ms
rtt min/avg/max/mdev = 0.339/0.389/0.488/0.073 ms
pi@raspberrypi4:~ $ ping 10.0.3.1 -c 3
PING 10.0.3.1 (10.0.3.1) 56(84) bytes of data.
64 bytes from 10.0.3.1: icmp_seq=1 ttl=64 time=0.291 ms
64 bytes from 10.0.3.1: icmp_seq=2 ttl=64 time=0.224 ms
64 bytes from 10.0.3.1: icmp_seq=3 ttl=64 time=0.299 ms

--- 10.0.3.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 91ms
rtt min/avg/max/mdev = 0.224/0.271/0.299/0.036 ms
pi@raspberrypi4:~ $ ping 10.0.4.1 -c 3
PING 10.0.4.1 (10.0.4.1) 56(84) bytes of data.
64 bytes from 10.0.4.1: icmp_seq=1 ttl=64 time=0.405 ms
64 bytes from 10.0.4.1: icmp_seq=2 ttl=64 time=0.332 ms
64 bytes from 10.0.4.1: icmp_seq=3 ttl=64 time=0.309 ms

--- 10.0.4.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 108ms
rtt min/avg/max/mdev = 0.309/0.348/0.405/0.046 ms
pi@raspberrypi4:~ $
```

Abbildung 5.11.: Ping Test As10

Es folgen weitere Messungen mit Kommandowerkzeugen. Zuerst folgen drei Messungen mit dem `ping` Werkzeug. Die Messung erfolgt von AS10 aus. Das Ergebnis der Messung ist in Abbildung 5.11 dargestellt, die die entsprechenden Ausgaben enthält.

Alle vier gewählten Ziele, in dem Fall AS20, AS30 und AS40 sind erreichbar. Es kam zu keinem Paketverlust, zu sehen an „0% packet loss“ bei allen drei Messungen.

Als nächstes werden Messungen mit dem `traceroute` Werkzeug gestartet. Die Abbildungen 5.12 und 5.13 stellen diese Messungen dar. In Abbildung 5.12 ist die Ausgabe des `traceroute` Tests, ausgeführt von AS10 aus, sowie in Abbildung 5.13 der ausgeführte Test von AS30 abgebildet. Zu sehen sind die verbundenen Router auf den Wegen zwischen den jeweiligen Zielen. Als Ziele wurden die jeweils anderen Netzwerke im Verbund gewählt. Für AS10 stellt das das AS20 (`10.0.2.1`), AS30 (`10.0.3.1`) und AS40 (`10.0.4.1`) dar. Für AS20 gilt entsprechend das gleiche, mit AS10 (`10.0.1.1`) statt AS20 als Ziel.

```
pi@raspberrypi4:~ $ traceroute 10.0.2.1
traceroute to 10.0.2.1 (10.0.2.1), 30 hops max, 60 byte packets
 1  10.0.2.1 (10.0.2.1)  0.543 ms  0.673 ms  0.553 ms
pi@raspberrypi4:~ $ traceroute 10.0.3.1
traceroute to 10.0.3.1 (10.0.3.1), 30 hops max, 60 byte packets
 1  10.0.3.1 (10.0.3.1)  0.294 ms  0.140 ms  0.098 ms
pi@raspberrypi4:~ $ traceroute 10.0.4.1
traceroute to 10.0.4.1 (10.0.4.1), 30 hops max, 60 byte packets
 1  10.0.4.1 (10.0.4.1)  0.419 ms  0.408 ms  0.433 ms
pi@raspberrypi4:~ $
```

Abbildung 5.12.: Traceroute Test AS10

```
root@orangepizeroplus:~# traceroute 10.0.1.1
traceroute to 10.0.1.1 (10.0.1.1), 30 hops max, 60 byte packets
 1  10.0.1.1 (10.0.1.1)  0.284 ms  0.186 ms  0.123 ms
root@orangepizeroplus:~# traceroute 10.0.2.1
traceroute to 10.0.2.1 (10.0.2.1), 30 hops max, 60 byte packets
 1  10.0.2.1 (10.0.2.1)  0.475 ms  0.433 ms  0.422 ms
root@orangepizeroplus:~# traceroute 10.0.4.1
traceroute to 10.0.4.1 (10.0.4.1), 30 hops max, 60 byte packets
 1  10.0.4.1 (10.0.4.1)  0.674 ms  0.858 ms  0.759 ms
root@orangepizeroplus:~#
```

Abbildung 5.13.: Traceroute Test AS30

Als abschließender Test wird eine Messung mit der `iperf` Software durchgeführt. Damit lässt sich der maximale Datendurchsatz (Bandbreite) der Ethernet-Schnittstelle messen. Zur Messung ist die Bereitstellung des Dienstes auf einem System nötig. AS10 stellt, in Abbildung 5.14 zu sehen, diesen Dienst mit `iperf -s` einen Server bereit. Alle drei weiteren Systeme verbinden sich daraufhin mit dem Server und führen eine Messung durch. Zu sehen ist die Ausgabe der entsprechenden Messung. Es verbinden

sich drei Klienten mit dem Server. Die IP-Adressen der verbundenen Klienten sowie die Datentransferrate und die Bandbreite pro Sekunde werden angezeigt. In Abbildung 5.15 ist die Seite des Klienten auf AS20 dargestellt. Dort wird ebenfalls das `iperf` Kommando ausgeführt, mit der IP-Adresse des Servers `10.0.1.1` und dem Argument „`-c`“, um sich mit der IP-Adresse als Klient zu verbinden. Zu sehen ist eine Rückmeldung des Programms, dass es eine TCP Verbindung aufgebaut und den Test durchgeführt hat. Die Testdauer, die Transferrate und die Bandbreite der Verbindung stimmen mit der Ausgabe auf dem Server überein.

```
pi@raspberrypi4:~ $ iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 10.0.1.1 port 5001 connected with 10.0.1.20 port 39016
[  5] local 10.0.1.1 port 5001 connected with 10.0.4.1 port 53790
[  6] local 10.0.1.1 port 5001 connected with 10.0.3.1 port 41808
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0-10.0 sec   350 MBytes   293 Mbits/sec
[  5]  0.0-10.0 sec   352 MBytes   294 Mbits/sec
[  6]  0.0-10.0 sec   605 MBytes   506 Mbits/sec
^Cpi@raspberrypi4:~ $
```

Abbildung 5.14.: Iperf Test AS10

```
pi@raspberrypi3:~ $ iperf -c 10.0.1.1
------------------------------------------------------------
Client connecting to 10.0.1.1, TCP port 5001
TCP window size: 43.8 KByte (default)
------------------------------------------------------------
[  3] local 10.0.1.20 port 39016 connected with 10.0.1.1 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3] 0.0000-10.0046 sec   350 MBytes   294 Mbits/sec
pi@raspberrypi3:~ $
```

Abbildung 5.15.: Iperf Test AS20

Die Messungen stellen den Abschluss dieses Kapitels dar.

6. Bewertung

In diesem Kapitel folgt die Bewertung der Messungen und der Implementierung des Versuchsaufbaus im Allgemeinen sowie die Evaluierung des Designs im Hinblick auf die Fragestellung dieser Bachelorthesis. Die bereits bestehende Lösung, die in Kapitel drei beleuchtet wurde, wird verglichen und anhand der vorgesehenen Nutzung einer Lehrveranstaltung bewertet. Die eingesetzten Geräte werden im Kontext zum Versuchsaufbau bewertet.

6.1. Bewertung der Messungen

Die Ausgaben des Kommandos in Abbildung 5.7 lassen auf eine stabile BGP-Verbindung zwischen den jeweiligen AS schließen. Die gewünschte Anzahl an Nachbar-AS wurde erreicht und korrekt von FRRouting ausgegeben. Die IP-Adressen der Nachbar Router sind ebenfalls, wie im Design konstruiert und wie in der FRRouting Konfiguration voreingestellt, korrekt dargestellt. Ebenfalls sind die AS-Nummern der jeweiligen AS korrekt empfangen. Daher lässt sich auf eine korrekte Konfiguration für FRRouting schließen, die die Netzwerk Topologie aus Kapitel vier umsetzt. An der Dauer der etablierten BGP-Sitzungen, wie in Abbildung 5.7 angezeigt, lässt sich ablesen, dass die Sitzungen nicht unterbrochen werden und bei einem Neustart des Gerätes oder bei Neustart des FRRouting-Dienstes wieder korrekt aufgebaut werden. Die Anzahl an Präfixen, die empfangen und gesendet werden, stimmen ebenfalls mit der Anzahl an Netzwerken im Verbund überein.

Abbildung 5.8 zeigt die empfangenen Präfixe beziehungsweise die IP-Adressbereiche der Nachbar AS. Diese stimmen ebenfalls mit den auf den jeweiligen Geräten eingestellten IP-Adressbereichen überein. Die Tatsache, dass der jeweilige IP-Adressbereich des AS nur in der jeweiligen lokalen FRRouting Konfiguration vorkommt, zeigt, dass das Präfix korrekt über BGP an die Nachbar-AS verteilt wird. Die Informationen sind daher korrekt empfangen worden. Die Anzahl der verschiedenen AS-Pfade beträgt für jedes

Nachbar-AS drei, hervorgerufen durch die FRRouting Konfiguration, bei der die Präfixe an alle Nachbarn verteilt werden. Dies lässt ebenfalls auf eine korrekte Konfiguration und ein korrektes Verhalten während den BGP-Sitzungen schließen. Es ist sogar in diesem Design gewünscht und zeigt so alternative Routen für jedes AS. Ebenfalls werden die Routen als „beste“ Routen markiert, die den kürzesten AS-Pfad besitzen. An diesen Ausgaben lässt sich erkennen, dass die Weitergabe der Routen-Informationen und die BGP-Sitzungen korrekt funktionieren.

Die in Abbildung 5.9 und Abbildung 5.10, durch BGP empfangene, dargestellten Routen werden im FRRouting korrekt dargestellt und in das System eingepflegt. Die Wahl der besten Routen zeigt allerdings eine Anomalie. So werden für die AS30 und AS40 nicht die durch BGP empfangen Routen, sondern die direkt verbundenen (mit einem „C“ markierten) Routen vom System ausgewählt. Für AS20 wird die BGP Route ausgewählt. Eine Erklärung könnte sein, dass in der FRRouting Konfiguration die IP-Adresse mit der IP-Adresse der Geräte übereinstimmt und die IP-Adresse bereits durch den Ethernet Switch empfangen wird. Im Vergleich dazu steht die Ausgabe vom AS40 Router. Hier werden zwei, über BGP empfangene, Routen ausgewählt. Die direkt verbundenen Routen für AS10 und AS20 werden nicht angezeigt. Es könnte sein, dass beim Starten eines Gerätes die Routen zufällig entweder direkt verbundene oder BGP Routen ausgewählt werden.

Die Messungen, die mit Hilfe der Kommandozeilenwerkzeuge `ping` und `traceroute` durchgeführt worden sind, zeigen die vollständige Erreichbarkeit für alle Netzwerke. Die mit dem `ip` Werkzeug konfigurierten IP-Adressen können erreicht und die Router der jeweiligen AS im `traceroute` Weg korrekt dargestellt werden, wie Abbildung 5.11, 5.12 und 5.13 zeigen, da in der Ausgabe des `traceroute` Kommandos kein anderer Router beziehungsweise keine andere IP-Adresse eines Routers aufgeführt wurde. Die Ausgaben in Abbildung 5.14 zeigen, dass die Geräte ihre für BGP konfigurierten IP-Adressen verwenden, um sich im `iperf` Test mit dem Server zu verbinden.
Zusammengefasst zeigen die in Kapitel 5.6 durchgeführten Messungen und Kommandos zur Verifizierung eine korrekt aufgebaute Topologie zwischen allen Geräten beziehungsweise zwischen den konstruierten Autonomen Systemen.

6.2. Bewertung des Versuchsaufbaus

Dieser Abschnitt bewertet die verwendeten Technologien, das gewählte Design sowie die Implementierung des Versuchsaufbaus.

Das gewählte Design stellt sich, im Kontext zur Definition eines Internet-Knotens, als erfolgreich dar. Die Verbindungen zwischen den konstruierten AS kann aufgebaut und aufrecht gehalten werden, wie die Messungen aus Kapitel fünf zeigen. Die Verbindungen über die Ethernet-Schnittstelle funktioniert wie erwartet und bleibt konstant. Über die gewählten IP-Adressen und AS-Nummern werden AS-Pfade erstellt, die von allen Geräten an alle weiteren Teilnehmer verteilt werden. Die Anforderungen an den Versuchsaufbau, die in Kapitel vier beschrieben werden, konnten in der Implementierung vollständig umgesetzt werden. Die Implementierung konnte mit Hilfe der gut dokumentierten FRRouting Software umgesetzt werden. Jedoch erfordert die Einrichtung der Betriebssysteme eine Internetverbindung, diese musste durch den zusätzlich eingesetzten Router hergestellt werden.

Der eingesetzte Ethernet Switch funktioniert im Versuchsaufbau sehr gut. Die BGP-Verbindungen über das Gerät können wie erhofft aufgebaut werden. Zusätzlich verändert der Switch die Pakete nicht und ermöglicht eine ungehinderte Kommunikation zwischen den AS.

Der Router, der für die Verteilung der IP-Adressen der WLAN-Schnittstellen zuständig ist, ist in diesem Versuchsaufbau nicht von großer Wichtigkeit und ermöglicht nur die Verbindung per Konsole über die WLAN-Schnittstelle. Dieser stellt bei einer Simulation eines Internet-Knoten keine relevanten Funktionen bereit und kann entsprechend weggelassen werden. Der Router ermöglicht die parallele Steuerung aller Einplatinencomputer im Verbund. Die Steuerung der Geräte könnte auch gegebenenfalls über die USB-Schnittstellen erfolgen.

Die eingesetzten Einplatinencomputer besitzen alle Funktionalitäten, um den Versuchsaufbau umzusetzen. Sie besitzen jeweils eine Ethernet-Schnittstelle, die die Kommunikation untereinander ermöglicht. Die verbaute WLAN-Schnittstelle eignet sich ideal zur Kommunikation und zur Fernsteuerung. Die in den Raspberry Pi verbauten HDMI-Schnittstellen ermöglichen zusätzlich die visuelle Darstellung, falls erforderlich,

und bieten durch USB-Schnittstellen die Möglichkeit, Eingabegeräte anzuschließen. Das SD-Karten Schubfach ermöglicht es, schnell mit verschiedenen SD-Karten verschiedene Betriebssystem-Abbilder auszutauschen und so verschiedene Distributionen zu testen.
Der Orange Pi Zero Plus jedoch bietet neben einer physischen Ethernet-Schnittstelle nur eine zusätzliche physische USB-Schnittstelle und kann ohne zusätzlichen Adapter, der die USB-Signale in beispielsweise HDMI umwandelt, nicht an einen Monitor oder eine Bildschirmausgabe angeschlossen werden. Die Steuerung muss daher zwingend über die Kommandozeile erfolgen. Die Handhabung der Raspberry Pi Geräte stellt sich daher als weitaus vielseitiger heraus.
Die Einplatinencomputer erfüllen die Anforderung für eine Demonstration und einen Versuchsaufbau: Durch ihre Größe können sie einfach transportiert werden. Sie sind durch das SD-Karten Schubfach individuell konfigurierbar und schnell einsatzbereit. Dabei sind sie leise im Betrieb, denn die verbauten Chips benötigen keine zusätzliche Kühlvorrichtung, die eine entsprechende Lautstärke abgeben könnte.
Durch den Ethernet-Port können alle Geräte miteinander kommunizieren. Die Routing-Software FRRouting stellt alle Funktionalitäten zur Verfügung, um eine BGP-Sitzung zwischen den konfigurierten Nachbar-Netzwerken aufzubauen.

FRRouting bietet alle Funktionen, um eine Inter-AS-Kommunikation aufzubauen. Die Software erfordert eine genaue Konfiguration der Nachbar-AS und eine genaue Syntax in der Konfigurationsdatei. Die Routing-Software stellt alle Funktionalitäten zur Verfügung, um eine dauerhafte BGP-Sitzung zwischen den konfigurierten Nachbar Netzwerken aufzubauen.

Die Einplatinencomputer stellen eine kostengünstige Alternative zu herkömmlichen Servern dar. Zwar besitzen die Einplatinencomputer nicht die Rechenleistung und die Arbeitsspeicher Größe des eingesetzten Servers im „Mini-Internet“ Projekt der ETH Zürich, jedoch reichen die verbauten Prozessoren und der Arbeitsspeicher vollkommen für die Demonstration der Inter-AS-Kommunikation in einem Vortrag oder einer Lehrveranstaltung. Jedoch benötigen die Einplatinencomputer zusätzliche Komponenten für einen Betrieb. Die benötigten zusätzlichen Netzteile zur Stromversorgung und die zusätzlichen SD-Karten erhöhen die Kosten, die nötig sind eine Simulation umzusetzen. Beim Hinzufügen von weiteren Geräten in den Verbund erhöhen sich die Kosten entsprechend.

Die Anwendungsfälle des in dieser Bachelorthesis aufgebauten Verbunds und das Projekt der ETH Zürich (siehe Kapitel drei) unterscheiden sich von einander, da das Projekt der ETH Zürich auf einen Kurs ausgelegt ist, bei dem die Teilnehmer aktiv neue Systeme konfigurieren und AS zusammenschalten. Der hier implementierte Verbund kann für Demonstrationszwecke und für den beispielhaften Fall einer Vorführung verwendet werden, nicht jedoch für mehrere Teilnehmer, die jeweils ein eigenes AS konfigurieren können.

7. Fazit

In dieser Bachelorthesis konnte gezeigt werden, dass aus einem Verbund aus Einplatinencomputern die Inter-AS-Kommunikation und damit ein Internet-Knoten simuliert werden kann. Die eingesetzten Geräte kommunizieren miteinander und tauschen Routing-Informationen miteinander aus. Durch den Einsatz von Routing-Software konnten BGP-Sitzungen etabliert und aufrecht gehalten werden. Die gewählte Topologie konnte erfolgreich umgesetzt und mit den Geräten abgebildet werden.
Der Verbund bietet eine gute Basis für eine eventuelle Lehrveranstaltung und könnte eingesetzt werden, um die Funktionen von BGP und der Inter-AS-Kommunikation in einem Vortrag darzustellen. Die eingesetzten Geräte sind in der Lage, die Vorgänge zu visualisieren und bieten die Möglichkeit, Testszenarien zu simulieren. Beispielsweise könnten automatisierte Vorgänge programmiert und so Testszenarien durchlaufen werden. Die verbundenen Netze könnten verschiedene Rollen übernehmen und mit den Teilnehmern des Verbunds auf ganz unterschiedliche Art und Weise kommunizieren.
Der verwendete Ethernet Switch stellt technisch gesehen einen Internet-Knoten dar, jedoch ohne Routing-Entscheidungen treffen zu können. Durch die Konfiguration eines vollvermaschten Netzwerkes aus vier AS ist die Anzahl an BGP-Sitzungen pro Teilnehmer hoch: Es sind $n - 1$ Verbindungen nötig, um alle Teilnehmer miteinander zu verbinden. Bei vier Geräten sind jeweils drei Verbindungen nötig. Erhöht man die Anzahl, steigt der Aufwand zur Konfigurierung entsprechend.
Der Einsatz eines Routen-Servers, der die benötigten Verbindungen zu allen Netzwerken herstellt und verwaltet, könnte die die Anzahl an benötigten Verbindungen für alle Teilnehmer auf jeweils eine Verbindung reduzieren, nämlich die Verbindung zum Routen-Server selbst. Der Server könnte Routen validieren, sie auf Richtigkeit überprüfen und so die Sicherheit des Internet-Knotens verbessern. Der Routen-Server müsste über ein eigenständiges System betrieben werden. Da im implementierten Versuchsaufbau keinerlei Überprüfung der Informationen von den verbundenen AS stattfindet, könnte sich der Versuchsaufbau dahingehend erweitern lassen, die Sicherheitsaspekte im Betrieb eines Internet-Knotens darzustellen.

Ebenfalls könnte der Testaufbau genutzt werden, um mit Hilfe von automatisierten Skripten die Funktionen eines Distributed-Denial-of-Service Angriffs, zu Deutsch wörtlich Dienstverweigerungsangriff, aufzuzeigen. Dabei können beispielsweise zwei AS die Schnittstelle eines Ziels auslasten, während ein drittes Gerät versucht dieses zu erreichen. Die Funktionen des BGP-Hijackings beziehungsweise Route-Hijacking können mit Hilfe des Versuchsaufbaus ebenfalls veranschaulicht werden. BGP-Hijacking ist ein Begriff, der eine Routen-Annoncierung beschreibt, bei der sich ein AS fälschlicherweise als ein anderes ausgibt, indem es Präfixe annonciert, die nicht zum eigenen AS gehören.
Abschließend können die Betriebssystemabbilder der Geräte in einer Cloud-Umgebung instanziiert werden und so beispielsweise Studenten ermöglichen, eigene Instanzen aufzubauen. Damit könnte zum Projekt der ETH aufgeschlossen werden, bei der die Teilnehmer eigene Konfigurationen erproben können.

Literatur

[1] Bernhard Ager u. a. “Anatomy of a Large European IXP”. In: *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*. SIGCOMM ’12. Helsinki, Finland: Association for Computing Machinery, 2012, 163–174. ISBN: 9781450314190. DOI: 10.1145/2342356.2342393. URL: https://doi.org/10.1145/2342356.2342393.

[2] *AMS-IX Amsterdam*. URL: https://www.ams-ix.net/ (besucht am 03. 12. 2021).

[3] *AMS-IX Statistik*. URL: https://www.ams-ix.net/ams/documentation/total-stats (besucht am 03. 12. 2021).

[4] *Armbian Dokumentation*. URL: https://docs.armbian.com/User-Guide_Getting-Started/ (besucht am 03. 12. 2021).

[5] Christian Baun. *Computernetze Kompakt*. Springer Vieweg, 2020.

[6] *Border Gateway Protocol (BGP)*. RFC 1163. Juni 1990. DOI: 10.17487/RFC1163. URL: https://rfc-editor.org/rfc/rfc1163.txt.

[7] Robert T. Braden. *Requirements for Internet Hosts - Communication Layers*. RFC 1122. Okt. 1989. DOI: 10.17487/RFC1122. URL: https://rfc-editor.org/rfc/rfc1122.txt.

[8] *DE-CIX*. URL: https://www.de-cix.net/de (besucht am 03. 12. 2021).

[9] *DE-CIX Frankfurt Statistik*. URL: https://www.de-cix.net/en/locations/frankfurt/statistics (besucht am 03. 12. 2021).

[10] *DE-CIX Globepeer Service*. URL: https://www.de-cix.net/de/services/globepeer (besucht am 03. 12. 2021).

[11] *DE-CIX Route Server Guide*. URL: https://www.de-cix.net/en/locations/frankfurt/route-server-guide (besucht am 18. 12. 2021).

[12] Klaus Dembowski. *Raspberry Pi – Das technische Handbuch*. Springer Vieweg, 2019.

[13] *EURO-IX 2020 Report*. URL: https://www.euro-ix.net/media/filer_public/cf/7c/cf7c8cb1-40c9-4e37-9d79-02b61ccc081e/ixp_report_2020_.pdf (besucht am 06. 12. 2021).

[14] Raspberry Pi Foundation. *Trustees' Report and financial statements 2020*. URL: https://static.raspberrypi.org/files/about/RaspberryPiFoundationReport2020.pdf (besucht am 27. 11. 2021).

[15] *FRRouting*. URL: https://frrouting.org (besucht am 01. 11. 2021).

[16] *FRRouting Installation*. URL: https://docs.frrouting.org/en/latest/setup.html (besucht am 19. 12. 2021).

[17] *FRRouting Systemvoraussetzungen*. URL: http://docs.frrouting.org/en/latest/overview.html#system-requirements (besucht am 03. 12. 2021).

[18] John A. Hawkinson und Tony J. Bates. *Guidelines for creation, selection, and registration of an Autonomous System (AS)*. RFC 1930. März 1996. DOI: 10.17487/RFC1930. URL: https://rfc-editor.org/rfc/rfc1930.txt.

[19] Thomas Holterbach u. a. "An Open Platform to Teach How the Internet Practically Works". In: *SIGCOMM Comput. Commun. Rev.* 50.2 (2020), 45–52. ISSN: 0146-4833. DOI: 10.1145/3402413.3402420. URL: https://doi.org/10.1145/3402413.3402420.

[20] *IXP Definition*. URL: http://www.ix-f.net/ixp-definition.html (besucht am 06. 12. 2021).

[21] *LINX*. URL: https://www.linx.net/ (besucht am 03. 12. 2021).

[22] *LINX Statistik*. URL: https://portal.linx.net/okta-login (besucht am 03. 12. 2021).

[23] William B. Norton. *The Internet Peering Playbook*. DRPeering Press, 2014.

[24] *Orange Pi*. URL: http://www.orangepi.org (besucht am 29. 11. 2021).

[25] *Quagga Routing Software Suite*. URL: https://www.quagga.net (besucht am 01. 11. 2021).

[26] *Raspberry Pi Software*. URL: https://www.raspberrypi.com/software/ (besucht am 29. 11. 2021).

[27] Yakov Rekhter, Susan Hares und Tony Li. *A Border Gateway Protocol 4 (BGP-4)*. RFC 4271. Jan. 2006. DOI: 10.17487/RFC4271. URL: https://rfc-editor.org/rfc/rfc4271.txt.

[28] Philipp Richter u. a. “Peering at Peerings: On the Role of IXP Route Servers”. In: *Proceedings of the 2014 Conference on Internet Measurement Conference*. IMC ’14. Vancouver, BC, Canada: Association for Computing Machinery, 2014, 31–44. ISBN: 9781450332132. DOI: 10.1145/2663716.2663757. URL: https://doi.org/10.1145/2663716.2663757.

[29] *RIPE Network Coordination Centre*. URL: https://www.ripe.net (besucht am 18. 11. 2021).

[30] Jesse Sowell. *Framing the Value of Internet Exchange Participation*. 2013. URL: http://dx.doi.org/10.2139/ssrn.2241841.

[31] *Zebra Routing Software*. URL: http://www.zebra.org (besucht am 01. 11. 2021).

A. dhcpcd.conf

```
1  # A sample configuration for dhcpcd.
2  # See dhcpcd.conf(5) for details.
3
4  # Allow users of this group to interact with dhcpcd via the control
       socket.
5  #controlgroup wheel
6
7  # Inform the DHCP server of our hostname for DDNS.
8  hostname
9
10 # Use the hardware address of the interface for the Client ID.
11 clientid
12 # or
13 # Use the same DUID + IAID as set in DHCPv6 for DHCPv4 ClientID as
       per RFC4361.
14 # Some non-RFC compliant DHCP servers do not reply with this set.
15 # In this case, comment out duid and enable clientid above.
16 #duid
17
18 # Persist interface configuration when dhcpcd exits.
19 persistent
20
21 # Rapid commit support.
22 # Safe to enable by default because it requires the equivalent
       option set
23 # on the server to actually work.
24 option rapid_commit
25
26 # A list of options to request from the DHCP server.
27 option domain_name_servers, domain_name, domain_search, host_name
```

```
28  option classless_static_routes
29  # Respect the network MTU. This is applied to DHCP routes.
30  option interface_mtu
31
32  # Most distributions have NTP support.
33  #option ntp_servers
34
35  # A ServerID is required by RFC2131.
36  require dhcp_server_identifier
37
38  # Generate SLAAC address using the Hardware Address of the
        interface
39  #slaac hwaddr
40  # OR generate Stable Private IPv6 Addresses based from the DUID
41  slaac private
42
43  # Example static IP configuration:
44  #interface eth0
45  #static ip_address=192.168.0.10/24
46  #static ip6_address=fd51:42f8:caae:d92e::ff/64
47  #static routers=192.168.0.1
48  #static domain_name_servers=192.168.0.1 8.8.8.8 fd51:42f8:caae:d92e
        ::1
49
50  # It is possible to fall back to a static IP if DHCP fails:
51  # define static profile
52  #profile static_eth0
53  #static ip_address=192.168.1.23/24
54  #static routers=192.168.1.1
55  #static domain_name_servers=192.168.1.1
56
57  # fallback to static profile on eth0
58  #interface eth0
59  #fallback static_eth0
60
61  ####
62  #
```

```
63 # Die nun folgenden Zeilen sind manuell eingetragen.
64 # Maximilian Willner 1052866
65 #
66 ###
67
68 # RASPBERRY PI 4
69
70 interface eth0
71 static ip_address=10.0.1.1/24
72 static routers=10.0.1.1
73 static domain_name_servers=10.0.1.1
74
75 ###
76
77 # RASPBERRY PI 3
78
79 interface eth0
80 static ip_address=10.0.2.1/24
81 static routers=10.0.2.1
82 static domain_name_servers=10.0.2.1
83
84 ###
85
86 # RASPBERRY PI 3 (2)
87
88 interface eth0
89 static ip_address=10.0.3.1/24
90 static routers=10.0.3.1
91 static domain_name_servers=10.0.3.1
92
93 ###
```

B. interfaces

```
1  source /etc/network/interfaces.d/*
2  # Network is managed by Network manager
3  auto lo
4  iface lo inet loopback
5
6  ###
7  # Orange Pi Zero Plus Schnittstellen Konfiguration
8  # Maximilian Willner 1052866
9  ###
10
11 auto eth0
12 iface eth0 inet static
13 address 10.0.3.1
14 netmask 255.255.255.0
15 gateway 10.0.3.1
16
17 #auto wlan0
18 #iface wlan0 inet dhcp
```

C. rc.local

```
1  #!/bin/sh -e
2  #
3  # rc.local
4  #
5  # This script is executed at the end of each multiuser runlevel.
6  # Make sure that the script will "exit 0" on success or any other
7  # value on error.
8  #
9  # In order to enable or disable this script just change the
       execution
10 # bits.
11 #
12 # By default this script does nothing.
13
14 # AS10 (Raspberry Pi 4):
15 ip address add 10.0.4.10/24 dev eth0:0
16 ip address add 10.0.3.10/24 dev eth0:1
17
18 # AS20 (Raspberry Pi 3):
19 ip address add 10.0.1.20/24 dev eth0:0
20 ip address add 10.0.4.20/24 dev eth0:1
21
22 # AS 30 (Orange Pi Zero Plus):
23 ip address add 10.0.2.30/24 dev eth0:0
24
25 # AS40 (Raspberry Pi 3 (2)):
26 ip address add 10.0.3.40/24 dev eth0:0
27
28 exit 0
```

D. FRRouting Installation

```
1  # add GPG key
2  curl -s https://deb.frrouting.org/frr/keys.asc | sudo apt-key add -
3
4  # possible values for FRRVER: frr-6 frr-7 frr-8 frr-stable
5  # frr-stable will be the latest official stable release
6  FRRVER="frr-stable"
7  echo deb https://deb.frrouting.org/frr $(lsb_release -s -c) $FRRVER
       | sudo tee -a /etc/apt/sources.list.d/frr.list
8
9  # update and install FRR
10 sudo apt update && sudo apt install frr frr-pythontools
```

E. daemons

```
1  # This file tells the frr package which daemons to start.
2  #
3  # Sample configurations for these daemons can be found in
4  # /usr/share/doc/frr/examples/.
5  #
6  # ATTENTION:
7  #
8  # When activating a daemon for the first time, a config file, even
       if it is
9  # empty, has to be present *and* be owned by the user and group "
       frr", else
10 # the daemon will not be started by /etc/init.d/frr. The
       permissions should
11 # be u=rw,g=r,o=.
12 # When using "vtysh" such a config file is also needed. It should
       be owned by
13 # group "frrvty" and set to ug=rw,o= though. Check /etc/pam.d/frr,
       too.
14 #
15 # The watchfrr, zebra and staticd daemons are always started.
16 #
17 bgpd=yes
18 ospfd=no
19 ospf6d=no
20 ripd=no
21 ripngd=no
22 isisd=no
23 pimd=no
24 ldpd=no
25 nhrpd=no
```

```
26  eigrpd=no
27  babeld=no
28  sharpd=no
29  pbrd=no
30  bfdd=no
31  fabricd=no
32  vrrpd=no
33  pathd=no
34
35  #
36  # If this option is set the /etc/init.d/frr script automatically
        loads
37  # the config via "vtysh -b" when the servers are started.
38  # Check /etc/pam.d/frr if you intend to use "vtysh"!
39  #
40  vtysh_enable=yes
41  zebra_options="  -A 127.0.0.1 -s 90000000"
42  bgpd_options="   -A 127.0.0.1"
43  ospfd_options="  -A 127.0.0.1"
44  ospf6d_options=" -A ::1"
45  ripd_options="   -A 127.0.0.1"
46  ripngd_options=" -A ::1"
47  isisd_options="  -A 127.0.0.1"
48  pimd_options="   -A 127.0.0.1"
49  ldpd_options="   -A 127.0.0.1"
50  nhrpd_options="  -A 127.0.0.1"
51  eigrpd_options=" -A 127.0.0.1"
52  babeld_options=" -A 127.0.0.1"
53  sharpd_options=" -A 127.0.0.1"
54  pbrd_options="   -A 127.0.0.1"
55  staticd_options="-A 127.0.0.1"
56  bfdd_options="   -A 127.0.0.1"
57  fabricd_options="-A 127.0.0.1"
58  vrrpd_options="  -A 127.0.0.1"
59  pathd_options="  -A 127.0.0.1"
60
61  # configuration profile
```

```
62 #
63 #frr_profile="traditional"
64 #frr_profile="datacenter"
65
66 #
67 # This is the maximum number of FD's that will be available.
68 # Upon startup this is read by the control files and ulimit
69 # is called.  Uncomment and use a reasonable value for your
70 # setup if you are expecting a large number of peers in
71 # say BGP.
72 #MAX_FDS=1024
73
74 # The list of daemons to watch is automatically generated by the
       init script.
75 #watchfrr_options=""
76
77 # To make watchfrr create/join the specified netns, use the
       following option:
78 #watchfrr_options="--netns"
79 # This only has an effect in /etc/frr/<somename>/daemons, and you
       need to
80 # start FRR with "/usr/lib/frr/frrinit.sh start <somename>".
81
82 # for debugging purposes, you can specify a "wrap" command to start
        instead
83 # of starting the daemon directly, e.g. to use valgrind on ospfd:
84 #   ospfd_wrap="/usr/bin/valgrind"
85 # or you can use "all_wrap" for all daemons, e.g. to use perf
       record:
86 #   all_wrap="/usr/bin/perf record --call-graph -"
87 # the normal daemon command is added to this at the end.
```

F. FRRouting services

```
1  Zebrasrv 2600/tcp  # zebra service
2  zebra  2601/tcp  # zebra vty
3  ripd  2602/tcp  # RIPd vty
4  ripngd  2603/tcp  # RIPngd vty
5  ospfd  2604/tcp  # OSPFd vty
6  bgpd  2605/tcp  # BGPd vty
7  ospf6d  2606/tcp  # OSPF6d vty
8  ospfapi  2607/tcp  # ospfapi
9  isisd  2608/tcp  # ISISd vty
10 babeld  2609/tcp  # BABELd vty
11 nhrpd  2610/tcp  # nhrpd vty
12 pimd  2611/tcp  # PIMd vty
13 ldpd  2612/tcp  # LDPd vty
14 eigprd  2613/tcp  # EIGRPd vty
15 bfdd  2617/tcp  # bfdd vty
16 fabricd  2618/tcp  # fabricd vty
17 vrrpd  2619/tcp  # vrrpd vty
```

G. AS10 frr.conf

```
1  frr version 8.0.1
2  frr defaults traditional
3  hostname raspberrypi4
4  log file /var/log/frr/bgpd.log
5  log stdout
6  no ip forwarding
7  no ipv6 forwarding
8  service integrated-vtysh-config
9  !
10 password pi
11 !
12 router bgp 10
13 bgp router-id 10.0.1.1
14 bgp bestpath as-path multipath-relax
15 neighbor peer peer-group
16 # AS 20
17 neighbor 10.0.1.20 remote-as 20
18 neighbor 10.0.1.20 peer-group peer
19 # AS 30
20 neighbor 10.0.3.1 remote-as 30
21 neighbor 10.0.3.1 peer-group peer
22 # AS 40
23 neighbor 10.0.4.1 remote-as 40
24 neighbor 10.0.4.1 peer-group peer
25 !
26 address-family ipv4 unicast
27 network 10.0.1.0/24
28 neighbor peer route-map peering-in in
29 neighbor peer route-map peering-out out
30 neighbor peer activate
```

```
31  exit-address-family
32  !
33  route-map upstream-out permit 100
34  set local-preference 10
35  !
36  route-map upstream-in permit 100
37  set local-preference 10
38  !
39  route-map peering-in permit 100
40  set local-preference 1000
41  !
42  route-map peering-out permit 100
43  set local-preference 1000
44  !
45  ip nht resolve-via-default
46  !
47  line vty
48  !
```

H. AS20 frr.conf

```
1  frr version 8.0.1
2  frr defaults traditional
3  hostname raspberrypi3
4  log file /var/log/frr/bgpd.log
5  log stdout
6  service integrated-vtysh-config
7  !
8  password pi
9  !
10 router bgp 20
11 bgp router-id 10.0.2.1
12 bgp bestpath as-path multipath-relax
13 neighbor peer peer-group
14 # AS 10
15 neighbor 10.0.1.1 remote-as 10
16 neighbor 10.0.1.1 peer-group peer
17 # AS 30
18 neighbor 10.0.2.30 remote-as 30
19 neighbor 10.0.2.30 peer-group peer
20 # AS 40
21 neighbor 10.0.4.1 remote-as 40
22 neighbor 10.0.4.1 peer-group peer
23 !
24 address-family ipv4 unicast
25 network 10.0.2.0/24
26 neighbor peer soft-reconfig inbound
27 neighbor peer route-map peering-in in
28 neighbor peer route-map peering-out out
29 neighbor peer activate
30 exit-address-family
```

```
31  !
32  route-map upstream-out permit 100
33  set local-preference 10
34  !
35  route-map upstream-in permit 100
36  set local-preference 10
37  !
38  route-map peering-in permit 100
39  set local-preference 1000
40  !
41  route-map peering-out permit 100
42  set local-preference 1000
43  !
44  #ip nht resolve-via-default
45  !
46  line vty
47  !
```

I. AS30 frr.conf

```
1  frr version 8.0.1
2  frr defaults traditional
3  hostname orangepizeroplus
4  log file /var/log/frr/bgpd.log
5  log stdout
6  service integrated-vtysh-config
7  !
8  password pi
9  !
10 router bgp 30
11 bgp router-id 10.0.3.1
12 bgp bestpath as-path multipath-relax
13 neighbor peer peer-group
14 # AS 10
15 neighbor 10.0.3.10 remote-as 10
16 neighbor 10.0.3.10 peer-group peer
17 # AS 20
18 neighbor 10.0.2.1 remote-as 20
19 neighbor 10.0.2.1 peer-group peer
20 # AS 40
21 neighbor 10.0.3.40 remote-as 40
22 neighbor 10.0.3.40 peer-group peer
23 !
24 address-family ipv4 unicast
25 network 10.0.3.0/24
26 neighbor peer soft-reconfig inbound
27 neighbor peer route-map peering-in in
28 neighbor peer route-map peering-out out
29 neighbor peer activate
30 exit-address-family
```

```
31  !
32  route-map upstream-out permit 100
33  set local-preference 10
34  !
35  route-map upstream-in permit 100
36  set local-preference 10
37  !
38  route-map peering-in permit 100
39  set local-preference 1000
40  !
41  route-map peering-out permit 100
42  set local-preference 1000
43  !
44  ip nht resolve-via-default
45  !
46  line vty
47  !
```

J. AS40 frr.conf

```
1  frr version 8.0.1
2  frr defaults traditional
3  hostname raspberrypi
4  log file /var/log/frr/bgpd.log
5  log stdout
6  service integrated-vtysh-config
7  !
8  password pi
9  !
10 router bgp 40
11 bgp router-id 10.0.4.1
12 bgp bestpath as-path multipath-relax
13 neighbor peer peer-group
14 neighbor 10.0.3.1 remote-as 30
15 neighbor 10.0.3.1 peer-group peer
16 neighbor 10.0.4.10 remote-as 10
17 neighbor 10.0.4.10 peer-group peer
18 neighbor 10.0.4.20 remote-as 20
19 neighbor 10.0.4.20 peer-group peer
20 !
21 address-family ipv4 unicast
22 network 10.0.4.0/24
23 neighbor peer soft-reconfiguration inbound
24 neighbor peer route-map peering-in in
25 neighbor peer route-map peering-out out
26 exit-address-family
27 !
28 route-map upstream-out permit 100
29 set local-preference 10
30 !
```

```
31  route-map upstream-in permit 100
32  set local-preference 10
33  !
34  route-map peering-in permit 100
35  set local-preference 1000
36  !
37  route-map peering-out permit 100
38  set local-preference 1000
39  !
40  line vty
41  !
```