LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN

MNM
TEAM
MUNICH NETWORK MANAGEMENT TEAM

INSTITUT FÜR INFORMATIK DER LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN
LEHRSTUHL FÜR KOMMUNIKATIONSSYSTEME UND SYSTEMPROGRAMMIERUNG
PROF. DR. D. KRANZLMÜLLER

PD DR. V. DANCIU

Skriptum zu den einführenden Sitzungen des Hauptseminars

PROGRAMMIERBARE NETZE

Sommersemester 2016

Inhaltsverzeichnis

1 Einleitung

Programmierbare Netze bzw. *Software Defined Networks* (*SDN*) werden als neue, alternative Netzarchitektur vorgestellt, die besser geeignet sein soll, die heutigen Anforderungen an Netze und ihre Dienste zu erfüllen. Die Verschärfung der Anforderungen an das globale Internet entstammt der Veränderung seines Nutzungsprofils: in seinen Anfängen ein Forschungsnetz (mit militärem Hintergrund) ist das Internet heute ein allgemeines Werkzeug geworden — ohne jedoch grundlegende Veränderungen seiner Architektur oder seiner Protokolle zu erfahren. Dem enspringt das Spannungsfeld, das in Abbildung 1.1 skizziert ist.

Diese Schrift fasst die technischen Gegebenheiten des heutigen globalen Internet auf der Basis von Modellen zusammen, als Motivation für die genauere Betrachtung von programmierbaren Netzen.

Die Dienste, die das Netz heute erbringt sind weit jenseits der ursprünglichen Basisdienste wie Email und Dateiübertragung per FTP aufgefächert. Hinzu kommen Dienste, die früher in speziellen Netzen erbracht wurden, z.B. Telefonie, Fernsehen, und Videokonferenzen. Von etwa 400 Knoten im Jahr 1983[Han06] wuchs das Netz durch seine Kommerzialisierung so sehr, dass der IPv4-Addressraum 2010 als erschöpft gemeldet wurde und die Anzahl der BGP-Routeneinträge — also die Anzahl der verschiedenen Netze, die im Internet erreichbar sind — hat eine halbe Million überschritten (vgl. Abbildung 1.2). Durch die Netzanbindung vieler traditionell isoliert ausgeführter Applikationen und der Entstehung neuer netzbasierter Dienste begünstigt wird die Diensterbringung in die Rechenzentren zentralisiert.

Die technologische Entwicklung des globalen Internet ist sehr langsam. Akzeptabel funktionierende Protokolle können nur aufgrund schwerwiegender Probleme oder aber besonderer kommerzieller Vorteile erweitert oder zu ersetzt werden. [Han06].

Netzkomponenten (Switches, Router, ...) haben in der IT-Landschaft eine Sonderstellung: anders als bei Rechnern werden sie als Spezialkomponenten, bei denen Hardware und Software

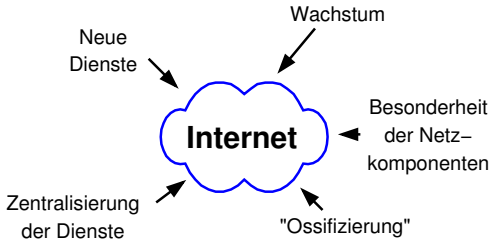

Abbildung 1.1: Einflüsse auf die technologische Entwicklung

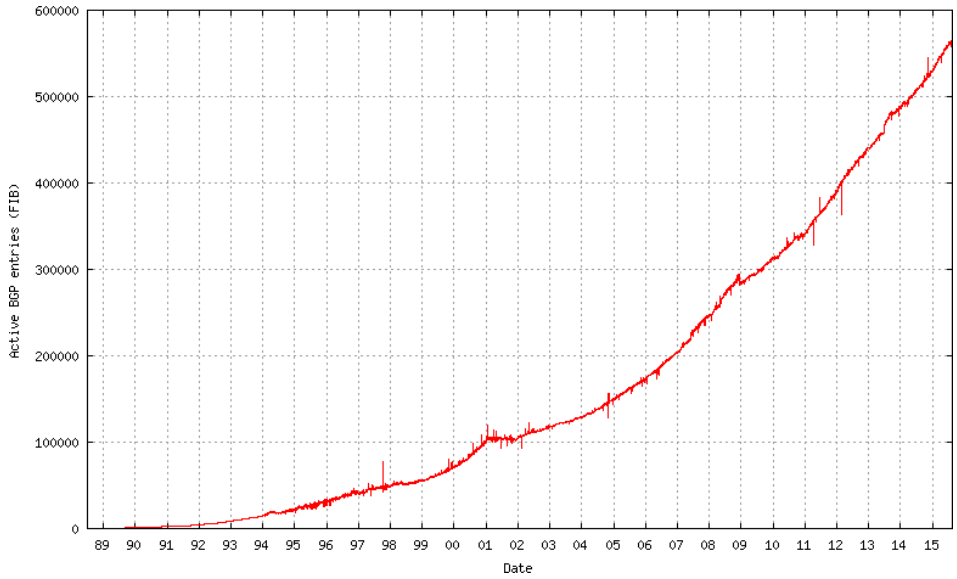

Abbildung 1.2: Wie viele Netze gibt es im Internet? Das Bild zeigt die Anzahl aktiver BGP-Routen (http://bgp.potaroo.net/as2.0/bgp-active.html, 2015-08-17)

stark voneinander abhängig sind vertrieben. Eine freie Wahl der Systemsoftware (und damit der Funktionen, die die Komponenten ausführen) ist — anders als etwa bei Rechnern mit PC-Architektur — nicht gegeben. Somit ist die Entwicklung der Funktionen dieser Komponenten von der Entwicklungslinie der Hersteller der Geräte abhängig.

Vor diesem Hintergrund existieren zwei Strategien zur Veränderung der Technologien und Architektur des Internet:

- Evolution, bei der eine Reihe kleiner Veränderungen basierend auf dem akzeptierten Zustand durchgeführt werden
- Revolution (auch *clean slate*), d.h. eine Neukonzeptionierung des Internet angepasst an die neuen Anforderungen.

SDN (und auch NFV) stammen aus der Forschung der *clean slate* Anhänger.

Die Anforderungen an das Netz, die sich von den ursprünglichen Entwurfsannahmen des Internet entfernen gemeinsam mit der Vermehrung der in Rechenzentren bereitgestellten Dienste führten zu der Konzeption programmierbarer Netze, deren Anwendung zunächst innerhalb der Rechenzentren — also primär intra-AS — vorgesehen ist.

In Kapitel 2 betrachten wir aktuelle Netze aus einer anderen als die bereits aus Vorlesungen bekannte ISO-OSI-Sicht. Diese Betrachtung versteht sich als Zugang zu der Einführung in programmierbaren Netzen bzw. SDN in Kapitel 3 und der Virtualisierung von Netzfunktionen (*Network Function Virtualization* (*NFV*)) in Kapitel 4.

2 Über die ISO-OSI-Brille geschielt

Hinführend betrachten wir zunächst ein von der Sicht der bekannten ISO-OSI-Architektur abweichendes Modell von Netzen und ihren Komponenten.

2.1 Netzkomponenten – der Status Quo

Das Netz enthält Koppelkomponenten verschiedener Schichten, insbesondere Brücken bzw. Switches der Sicherungsschicht (Schicht 2 des ISO-OSI-Modells) und Router in der Vermittlungschicht (Schicht 3). Eine Koppelkomponente führt Vermittlungsfunktionen aus (sie leitet Nachrichten weiter, an andere Koppelkomponenten oder an *Datenendeinrichtungen* (*DEE*) wie z.B. Rechner oder Speicherkomponenten). Ferner beherrscht eine Netzkomponente in der Regel autonome Managementfunktionen (z.B. Wegewahl) und bietet Schnittstellen für das Auslösen von Managementoperationen durch Administratoren an (z.B. SNMP, Managementkonsolen).

Der Begriff „Netzkomponenten“ umfasst im weiteren Text die Koppelkomponenten, also die Knoten des Netzes (hautptsächlich Switches und Router). Endpunkte bzw. DEE bezeichnen die nicht vermittelnden Rechner (und andere Geräte), die Teil des Netzes sind.

Im ISO-OSI-Modell werden Netzkomponenten nach den Modellschichten, die sie implementieren eingeordnet. Abbildung 2.1 zeigt einen Switch, der Bitübertragungs- und Sicherungsschicht implementiert sowie einen Router, der zusätzlich die Vermittlungsschicht umsetzt. Die Komponenten werden der höchsten Schicht zugeordnet, die sie umsetzen: ein Router ist also eine Komponente der Vermittlungsschicht, ein Switch eine der Sicherungsschicht.

Das OSI-Modell betrachtet jedoch nicht die schichtübergreifenden Funktionen, die reale Netzkomponenten erfüllen müssen. Solche Funktionen finden sich in mehreren Schichten. Beispielsweise führen Ethernet-Switches eine Form der Vermittlung durch, die wie bei Routern, Wegewahlentscheidungen voraussetzt: sie merken sich die Zuordnung von Ports zu MAC-Adressen und leiten gezielt eingehende Nachrichten weiter.

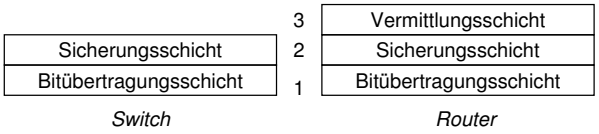

Abbildung 2.1: Switch und Router im OSI-Modell

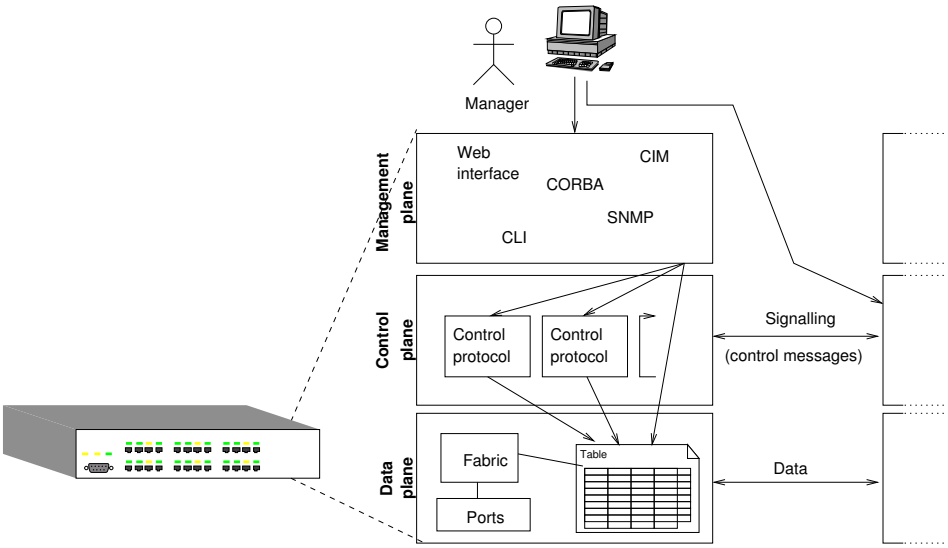

Abbildung 2.2: Beziehungen der Funktionen einer Netzkomponente

2.1.1 Ebenenarchitektur

Eine alternative Charakterisierung von Netzkomponenten (und somit des Netzes an sich) ist nach Ebenen möglich. Dabei werden genau die erwähnten Funktionen betrachtet, die unabhängig von dem Komponententyp vorhanden sind.

Diese Funktionen können drei Ebenen zugeordnet werden (Abbildung 2.2), der

1. *data plane* (auch *forwarding plane*), die insbesondere die Funktionen der Vermittlung beinhaltet; der
2. *control plane* für die Steuerungsfunktionen (autonome Managementfunktionen); und der
3. *management plane* für die Managementeingriffe durch Administratoren.

Die Vermittlung in der *data plane* (auch: *forwarding plane*) geschieht für gewöhnlich auf der Basis einer Vorschrift, z.B. einer Tabelle, die einer Nachricht ein direkt erreichbares Ziel zuordnet. Zu den Elementen dieser Ebene gehören also diese Vorschrift sowie die Ein-/Ausgabeschnittstellen (Ports) der Komponente.

In der *control plane* sind Wegewahl- und Kontrollprotokolle angesiedelt, die einen Austausch von Managementinformationen zwischen den Netzkomponenten ermöglichen und so Änderungen der Vermittlungsvorschrift in der *data plane* bewirken können. Zum Beispiel bewirkt der Austausch komponentenlokalen Wissens über den Netzzustand mittelbar Änderungen an den Routingtabellen der einzelnen am Austausch beteiligten Router. Dabei kommen die bekannten Protokolle (*Routing Information Protocol* (*RIP*) [Hed88], *Open Shortest Path First* (*OSPF*) [Moy98], *Border Gateway Protocol* (*BGP*) [RLH06], etc) zum Einsatz.

Weitere Funktionen der *control plane* sind die Signalisierung (etwa mit *Internet Control*

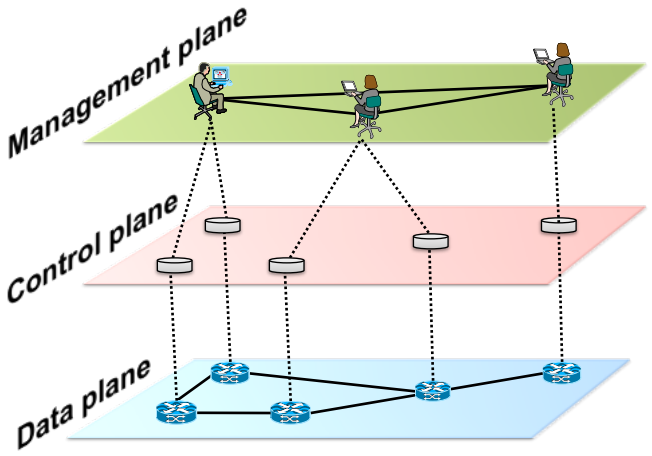

Abbildung 2.3: Ebenen über administrative Domänen hinweg (aus [KREV+15])

Message Protocol (*ICMP*) [Pos81]), die Erkennung zusammenhängender Datenflüsse (*flows*) in Routern oder etwa die Lernfunktion von Ethernet-Switches.

Darüber hinaus werden in dieser Ebene Statistiken über den Netzverkehr und andere, für Management und Betrieb nützliche Informationen (etwa Zustände und Zustandswechsel von Ports) gesammelt und der Managementebene zur Verfügung gestellt.

Die Funktionen der Managementebene (*management plane*) ermöglichen den Eingriff in den Betrieb durch Managementprotokolle: *Simple Network Management Protocol* (*SNMP*) (vgl. [FLRW03]), RPC (z.B. CORBA und Webservice-Schnittstellen) und Nutzerschnittstellen (Textkonsolen, graphische Schnittstellen). Die Interaktion mit der Managementebene bewirkt Änderungen am Zustand der *control plane* (z.B. Parameter von Routingprotokollen) und der *data plane* (z.B. direkte Änderungen an den Switching-Tabellen oder das Einrichten eines VLAN).

Abbildung 2.2 zeigt die Inhalte von und die Beziehungen zwischen den Ebenen in einer ganzheitlichen Sicht auf das Netz: eine Menge an Netzschnittstellen (*ports*), die in Wirklichkeit über die Koppelkomponenten verteilt sind und eine Wissensbasis für die Vermittlung; Kontroll- und Wegwahlprotokolle mittels derer Informationen über den Netzzustand propagiert werden, die zu Änderungen in Vermittlung führen, ferner Managementinformationen; schließlich eine Schnittstellenebene, die das Abrufen von Managementinformationen sowie den direkten Eingriff in die Konfiguration und den Betrieb des Netzes ermöglichen.

Alle Elemente der drei Ebenen sind in heutigen Netzen in jeder einzelnen Komponenten implementiert. Betrachtet man ein Verbundnetz in der Ebenensicht, ergibt sich ein Bild wie in Abbildung 2.3: die Vermittlungsebene enthält die vermittelnde Hardware, die Steuerungsebene enthält die für die Steuerung erforderlichen Daten und Verfahren und die Managementebene bildet die Schnittstelle für administrative Eingriffe.

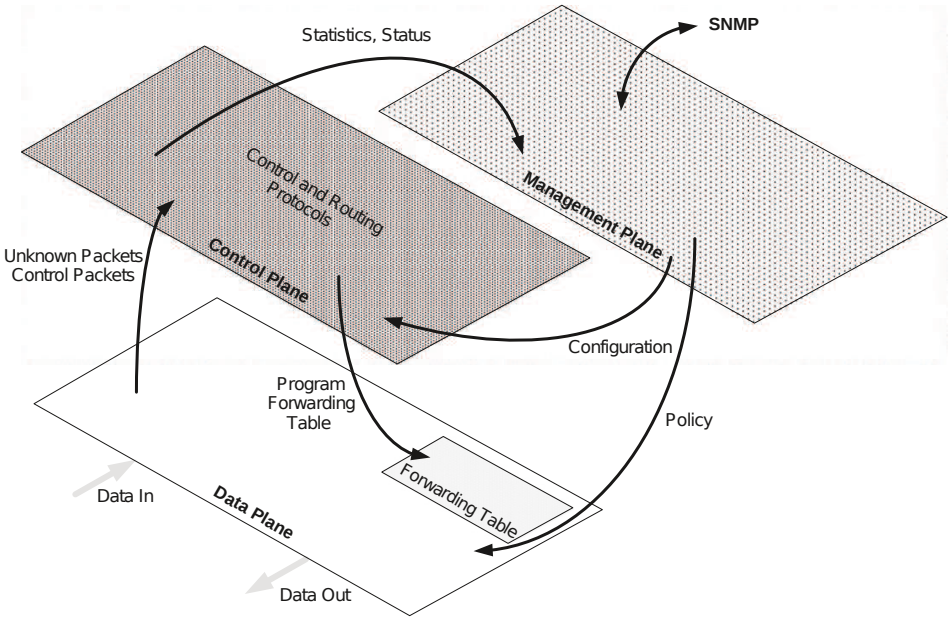

Abbildung 2.4: Inhalte und Beziehungen von *data, control* und *management plane* (aus [GB14])

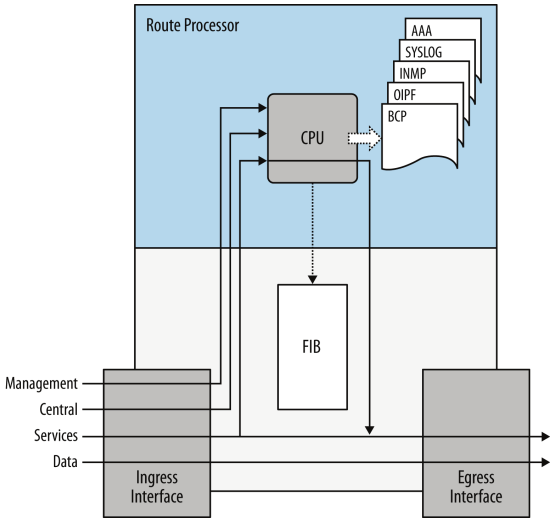

Abbildung 2.5: Aufteilung der Datenverarbeitung (aus [NG13])

Die rein funktionale Sichtweise unter Abstraktion von den Komponenten ist in Abbildung 2.4 illustriert: jede Ebene enthält die bereits beschriebenen Artefakte und interagiert mit den anderen Ebenen.

2.1.2 Aufbau und Funktion

Moderne Netzkomponenten benutzen unterschiedliche Hardware für die Weiterleitung in der Datenebene einerseits und der Ausführung der Steuerungsmechanismen und -protokolle in der Steuerungsebene andererseits. Abbildung 2.5 zeigt einen schematischen Aufbau eines Routers, der im oberen Teil die Hardwareunterstützung der Steuerungsebene, im unteren jene der Vermittlungsebene illustriert.

Der Router unterscheidet eingehende Nachrichten (an der Ingress-Schnittstelle) wie folgt: "normale" Nachrichten, deren Ziel bekannt ist werden anhand der *Forwarding Information Base* (*FIB*), also einer Weiterleitungstabelle des Routers, an eine der Egress-Schnittstellen ausgegeben.

Nachrichten eines Steuerungsprotokolls (z.B. ein Routingprotokoll) oder Managementnachrichten (z.B. SNMP) werden an den sog. *Route Processor* weitergegeben. Ihre Verarbeitung kann Veränderungen an der FIB bewirken.[1]

Eingehende Nachrichten werden in einem traditionellen Router den Verarbeitungsstufen in Abbildung 2.6 unterzogen: Entschlüsselung (z.B. bei IPSec-Nachrichten), Prüfung der Zugangskontrolle in das Netz, Prüfung von Dienstgüteregelungen (z.B. Begrenzung der Übertra-

[1] Der Originaltext von Nadeau und Gray [NG13] geht auf die Programmierung von ASIC aufgrund unerkannter Nachrichten ein. Für die Zwecke dieser Einführung genügt ein einfacheres Modell.

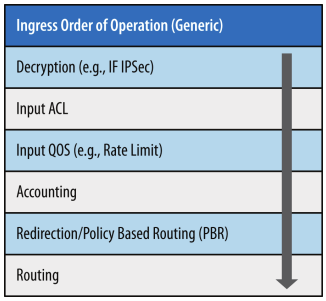

Abbildung 2.6: Funktionen auf eingehende Nachrichten ([NG13])

gungsrate), Kontoführung (z.B. für Abrechnung nach Volumen), Policy-based Routing (vgl. Abschnitt 2.1.7) und schließlich eine Vermittlungsentscheidung.[NG13]

Abbildung 2.7 zeigt den Informationsfluss, der zur Ermittlung der Vermittlungstabelle führt. Die Interaktionen zwischen den Instanzen verschiedener Protokolle im selben Gerät werden durch die CPU der Komponente durchgeführt. Jede weitere Protokollinteraktion (z.B. eine eingehende OSPF- oder BGP-Nachricht) kann zu einer erneuten Verarbeitung führen.

2.1.3 Entwurfsannahmen

Die dargestellte Bauweise autonomer Netzkomponenten ist eine Folge früher Konstruktionsannahmen, dass:

- das Netz aus unzuverlässigen Knoten und Endsystemen bestehe,
- Schicht-1-Verbindungen wenig robust seien,
- die Topologie des Netzes unbekannt sei (und daher dynamisch ermittelt werden müsse) und dass
- die Steuerung des Netzes nach technischen Kriterien (z.B. Erhaltung der Konnektivität) erfolge.

Vor diesem Hintergrund wirkt die Wahl von verteilten Verfahren und die verteilte Haltung der Steuerungs- und Managementdaten selbstverständlich, wie auch die lokale Entscheidungsfindung in jeder Netzkomponente auf der Basis von Signalisierungsdaten.

2.1.4 Nachteile

Die Autonomie der Netzkomponenten erfordert in jeder Komponente die Implementierung der Steuerungprotokolle und der Verfahren für die Entscheidungsfindung. Die Herstellungkosten für solche Netzkomponenten erhöhen sich dadurch, bedenkt man insbesondere, dass eine Vielzahl an Steuerungsprotokollen unterstützt werden müssen, um die Interoperabilität mit einer möglichst großen Anzahl an unterschiedliche Komponenten (andere Hersteller, andere Modelle, andere Auswahl an Steuerungsprotokollen durch den Netzbetreiber) zu erhalten.

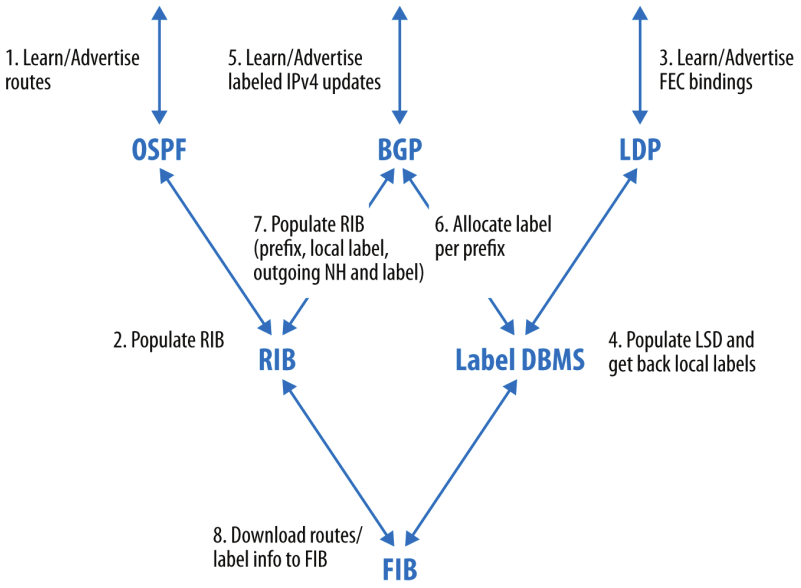

Abbildung 2.7: Informationsfluss zur Bestimmung der FIB (aus [NG13])

Die Wirkungsweise vieler Steuerungsprotokolle erfordert eine Konvergenzzeit, bis eine Änderung des Netzzustandes richtig erkannt und an jede Netzkomponente propagiert wurde. Beispiele sind etwa Routingprotokolle oder Protokolle zwischen Ethernetbrücken (etwa das Spanning Tree Protocol, STP). Eine Änderung am Netz zieht also eine Konvergenzphase nach sich, die bei nur seltenen Änderungen hinzunehmen wäre, bei hohen Änderungsfrequenzen allerdings den Betrieb stört. Aufgrunddessen musste etwa das ursprüngliche *Spanning Tree Protocol* (*STP*) durch das *Rapid Spanning Tree Protocol* (*RSTP*)[IEE04] ersetzt werden; auf der Schicht 3 wurde das RIP aufgrund seiner Konvergenzzeit bereits früh von OSPF verdrängt [Tan11].

Die Nutzung verteilter Verfahren erfordert Signalisierungsverkehr, dessen Umfang nicht nur von den genutzten Protokollen, sondern auch von der Häufigkeit der Steuerung abhängig ist. In manchen Fällen (siehe z.B. Referenz 8 bei [GB14], S. 14) kann der Signalisierungsverkehr einen erheblichen Anteil am Gesamtverkehr haben.

2.1.5 Protokollbedingte Einschränkungen

Manche Einschränkungen aktueller Netztechniken sind bedingt durch die eingesetzten Protokolle.

Die Ethernet-Spezifikation schreibt etwa vor, dass die Schicht-2-Topologie keine Kreise enthalten darf. In vielen Fällen werden aber redundante Leitungen verlegt, um im Störungsfall den Verkehr umlenken zu können. Um im Normalfall Kreise zu vermeiden, werden die Ports, an

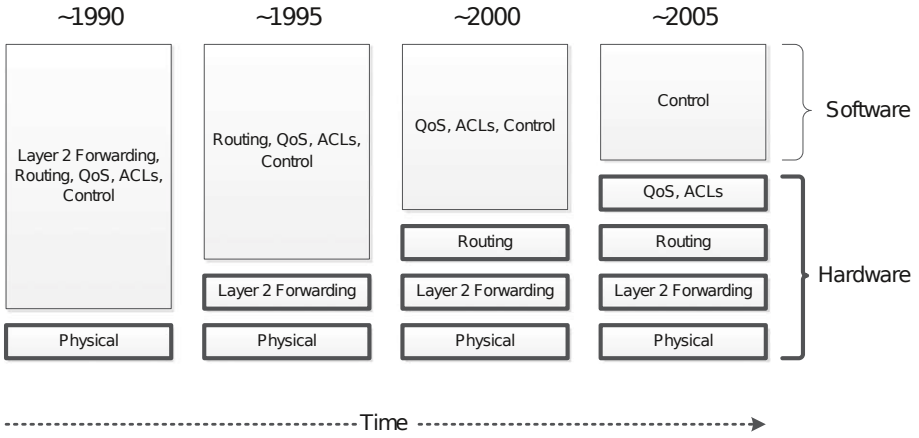

Abbildung 2.8: Zeitskala für die Aufnahme von Funktionen in die Hardware (aus [GB14])

denen die redundanten Leitungen angeschlossen werden, abgeschaltet – ihre Übertragungskapazität liegt also brach – und erst im Störungsfall zugeschaltet, manuell oder über RSTP. Diese Betriebsweise ist vorherrschend, obwohl Verbesserungsvorschläge standardisiert wurden, etwa *Shortest Path Bridging* (*SBP*) [FASA+12] oder RBridges mit *Transparent Interconnection of Lots of Links* (*TRILL*) [TP09], die beliebige Schicht-2-Topologien umsetzen könnten.

In der Vermittlungsschicht (OSI Schicht 3) werden verschiedene *Interior Gateway Protocols* (*IGP*) und *Exterior Gateway Protocol* (*EGP*) Versionen eingesetzt, wobei *Open Shortest Path First* (*OSPF*) [Moy98] und BGP vorherrschend sind. Das Wachstum des Internet sowie die Nutzung klassenloser Netzadressen (*Classless Inter-Domain Routing* (*CIDR*)) führen zu einer sehr großen Anzahl von Wegewahlangaben (Routen), die *jeder* Router verarbeiten muss.

2.1.6 Verlagerung der Funktionen in die Hardware

Die Funktionen des Netzes wurden im Zuge der technologischen Entwicklung zunehmend in der Hardware umgesetzt. Abbildung 2.8 zeigt diese Wandlung über die Zeit: während in frühen Netzen lediglich die Bitübertragung — gezwungenermaßen — in Hardware realisiert war, folgten zur Steigerung der Leistung der Netzkomponenten die Vermittlung in der Sicherungsschicht, die Wegewahl sowie Dienstgüte- und Zugriffskontrollmechanismen.

Zur Beschleuning der Vermittlung kamen zunächst ab ca. 1990 *Application Specific Integrated Circuit* (*ASIC*), also anwendungsspezifische integrierte Schaltungen (vulgo: Chips, speziell für die Vermittlung) und *Content Addressable Memory* (*CAM*), also assoziativer Speicher zum Einsatz. Heute werden zusätzlich *Field Programmable Gate Array* (*FPGA*), also programmierbare Schaltkreise sowie ternärer CAM (TCAM, ein assoziativer Speicher mit Maskierungsfunktion) zum Einsatz.[GB14] Steuerungs- und Managementprotokolle sowie die für sie erforderliche Signalisierung zwischen den Komponenten bleiben bis heute in Software realisiert.

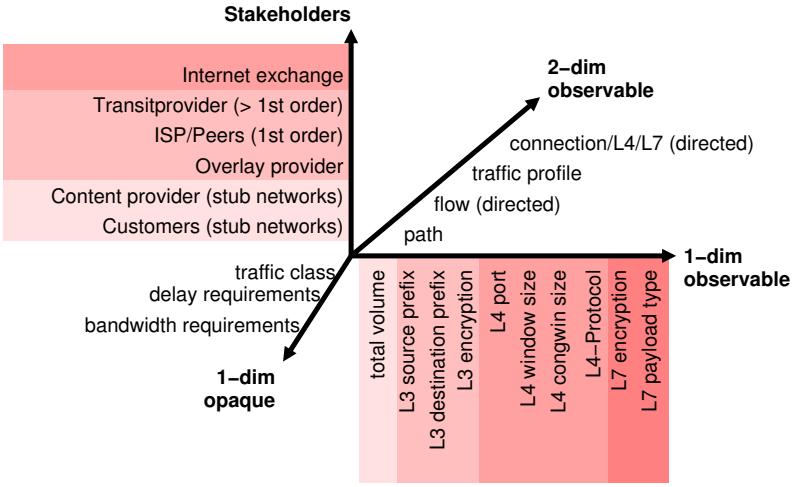

Abbildung 2.9: Dimensionenraum nicht-neutraler Vermittlung

2.1.7 Funktionen jenseits des Datentransportes

Das Ziel traditioneller Wegewahl im Internet ist die Erstellung und Erhaltung optimaler Pfade zwischen den Teilnehmern im Netz (Knoten und Endsysteme). Dabei sind die Identität der einzelnen Teilnehmer, deren Zugehörigkeit zu Organisationen, der Typ von Daten und die Inhalte des Netzverkehrs unerheblich: das Netz vermittelt neutral bezüglich dieser Attribute.

Policy-based Routing

Wegewahl und Vermittlung kann aber auch anhand von Richtlinien (engl. *Policies*) bestimmt werden, statt einer netzneutralen Behandlung des Verkehrs. Dabei kann unterschieden werden zwischen Qualitätsklassen, Dienstklassen (z.B. nach Anforderungen an das Netz, etwa Sprach- oder Videodienste versus Dateiübertragung), Sender- bzw. Empfängeridentitäten (z.B. „zahlende Kunden" versus „andere"). Sicherheitsmechanismen können ebenfalls in die Richtlinien für die Wegewahl und Vermittlung eingehen, etwa Zulassung/Verbot bestimmter Kommunikationsmuster durch netzbasierte *Intrusion Detection System* (*IDS*).

Solche richtlinienkonforme Wegewahl, *Policy-based Routing* (*PBR*) erfordert die Analyse des Netzverkehrs mit Bezug auf IP-Adressen, TCP-Ports, u.U. MAC-Adressen, Richtung, Datentyp der Inhalte, Verschlüsselung/Klartext, etc und trifft Wegewahl-/Vermittlungsentscheidungen anhand dieser Attribute bei eingehenden Nachrichten. Abbildung 2.9 skizziert den von Interessenten und von außerhalb des Netzes beobachtbaren Manipulationen aufgespannten Raum.

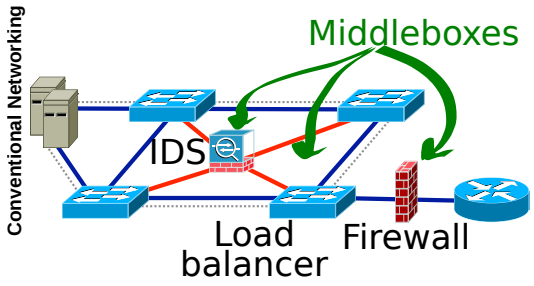

Abbildung 2.10: Netzfunktionen in selbständigen Netzkomponenten (basierend auf [KREV+15])

Middleboxes

Neben den Koppelkomponenten des Netzes (Switches, Router) werden in heutigen Netzen weitere Netzkomponenten eingesetzt, die Netzfunktionen jenseits der reinen Vermittlung von Nachrichten erfüllen. Dazu zählen sicherheitsrelevante Funktionen (durch Paketfilter/Firewalls), Analysefunktionen als Grundlage für *Traffic Management* (etwa durch *Deep Packet Inspection (DPI)*, also der Analyse der transportierten Daten und nicht nur der Protokollinformationen) und dienstspezifische Komponenten (Caches, Proxies, Filter der Anwendungsschicht, . . .).

Abbildung 2.10 zeigt die Anordnung einiger Middleboxes in einem kleinen Netz. Ihre Platzierung muss dabei ihrer Funktion entsprechen (z.B. die Firewall/Paketfilter am Ingress des Netzes), somit bestimmt der Einsatz von Middleboxes die Topologie des Netzes mit [KREV+15].

Diese Komponenten (*Middleboxes*) beeinflussen maßgeblich das Verhalten des Netzes, indem sie Richtlinien des Netzbetreibers umsetzen. Ihr Einsatz als dedizierte Komponenten erhöht die Komplexität des Netzes (z.B. werden eingehende Nachrichten aus einem benachbarten Netz nicht sofort an ihren Empfänger zugestellt, sondern zunächst an einen Paketfilter, dann an einen Proxy, usw).

2.2 Rechenzentren

Die Besonderheiten von SDN werden speziell für Netze in *Rechenzentren* (*RZ*) als geeignet betrachtet.

2.2.1 Charakterisierung von RZ-Netzen

Die Rechen- und Netzinfrastruktur in Rechenzentren weist Unterschiede zum allgemeinen Fall eines Netzes auf:

- sie wird in einer einzigen administrativen Domäne verwaltet,
- sie hat eine bekannte, zentral gemanagte Topologie

- Störungen treten relativ selten auf, eine zentrale Reaktion/Behebung ist möglich und erwünscht

Diese Eigenschaften unterscheiden sich von jenen, die (vgl. Abschnitt 2.1.3) dem Entwurf heutiger Netze zugrundelagen.

Durch die Verlagerung vormals lokal (bei den Nutzern) durchgeführten Rechenbetriebs in Rechenzentren werden die Anforderungen der Rechenzentren maßgeblich für die Bewertung (und damit der Konstruktion und dem Betrieb) von Infrastruktur. Wir betrachten nachstehend insbesondere die Anforderungen an die Netze in den Rechenzentren.

2.2.2 Anforderungen an RZ-Netze

Die Anforderungen an die Netze innerhalb moderner RZ ergeben sich aus aktuellen Betriebsszenarien. Hierzu gehören die schnelle Bereitstellung (verteilter) Recheninfrastruktur an die Kunden des RZ, die Zusicherung ausgehandelter Qualitätseingeschaften für diese Infrastruktur, und der Betrieb vieler solcher Infrastrukturinstanzen — also eine Mandantenfähigkeit des Management des RZ. “Cloud”-Dienste, also ad-hoc Bereitstellung von Ressourcen erfordern zudem die Abtretung administrativer Rechte an die Kunden/Dienstnehmer für die (virtuelle) Infrastruktur, die ihnen bereitgestellt wird. Diese Szenarien erfordern eine erhöhte Flexibilität im Betrieb der Netze im RZ.

Göransson und Black geben in [GB14] nachstehende Anforderungen moderner RZ-Netze an.

Automatisierung der Netzkonfiguration: Instanziierung und Zerstörung von Teilnetzen entsprechend der Bereitstellung von (virtuellen) Servern und Speicherelementen, ferner Anpassungen (Erweiterungen, Ausserbetriebnahme) von Teilen des Netzes.

Skalierbarkeit, zur Bedienung virtueller Komponenten. Durchbruch der Beschränkungen durch MAC-Tabellen bei Switches und der Anzahl möglicher VLANs (beschränkt durch geringe Breite der VLAN-ID).

Multipathing, d.h. die Nutzung aller vorhandenen, auch redundanter Pfade der Sicherungsschicht; somit Erweiterung der bisher durch die Standards vorgeschriebene Baumstruktur der Schicht-2-Topologie.

Mandantenfähigkeit: Management von Teilnetzen durch Mandanten, also Kunden des Betreibers, ohne unerwünschten Einfluss auf das Betreibernetz.

Netzvirtualisierung: Abstraktion der Mandantennetze vom Betreibernetz. Basis für die Bereitstellung rein virtueller Infrastruktur für Dienstnehmer (*Infrastructure as a Service*).

Aktuelle Forschung beschäftigt sich mit der Entwicklung programmierbarer Netze — SDN — die diesen Anforderungen gerecht werden können.

3 Software Defined Networks

Die Entwicklung *Software Defined Networks* (*SDN*) begründet sich durch die technische Entwicklung der Netzkomponenten (vgl. Abschnitt 2.1), durch die Managementfunktionalität, die in heutigen Netzen erwünscht und in Koppelkomponenten oder in Middleboxes umgesetzt wird (vgl. Abschnitte 2.1.7 und 2.1.7, ferner durch die Anforderungen von RZ-Netzen, deren Bedeutung in der aktuellen Phase der Infrastrukturkonsolidierung steigt (vgl. Abschnitt 2.2.

Die Grundidee von SDN besteht darin, eine Netzarchitektur unter anderen Annahmen zu schaffen, als zur Entwicklung des "ursprünglichen" globalen Internet galten. Dies beinhaltet einerseits den Verzicht auf grundsätzlich verteilte (und somit protokollgebundene) Entscheidungsfindung und andererseits die ausdrückliche Unterstützung von nicht-Vermittlungsfunktionen durch das Netz selbst.

3.1 Architektur

Im folgenden werden SDN anhand charakteristischer Merkmale abgegegrenzt und die Komponenten und Schnittstellen einer generischen SDN-Architektur besprochen.

3.1.1 Charakterisierende Eigenschaften

Für SDN werden nach Göransson und Black [GB14] vier definierende Eigenschaften beschrieben :

1. Die Trennung der Ebenen, insbesondere von *data plane* und *control plane*.

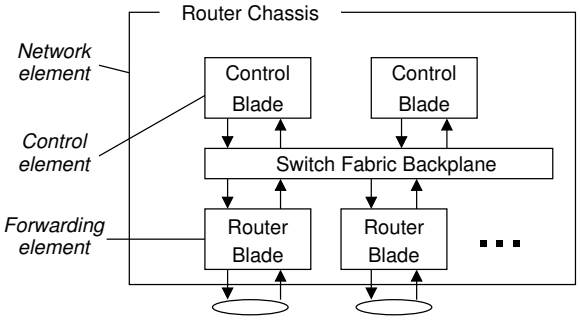

Abbildung 3.1: Früher Ansatz: ForCES (nach [YDAG04])

Die Aktionen innerhalb der *data plane* beschränken sich somit auf den Umgang mit eingehenden Nachrichten: vermitteln (*forward*), verwerfen (*drop*), verarbeiten (*consume*), vervielfältigen (*replicate*, z.B. als Broadcast). Verarbeitet werden Nachrichten, die besondere Behandlung durch höhere Ebenen (*control plane*) erfordern.

2. Vereinfachung der Geräte durch Beschränkung der Hardware auf die Implementierung der *data plane*.
3. Abstraktion vom verteilten Zustand, von der Vermittlung und der Konfiguration.

   Die Steuerung des Netzes ist unabhängig vom Ort der Speicherung eines Teils seines Zustandes, den spezifischen Vermittlungsmechanismen (eines Produkts/Herstellers) und der genauen Spezifikation einer Konfiguration (bei einem Produkt oder der Produktlinie eines Herstellers).
4. Interoperabilität und offene Schnittstellen

   Göransson und Black sehen dieses Kriterium als charakteristisch, wenn auch nicht technisch notwendig [GB14].

Die Sicht von Jain und Paul [JP13] definiert SDN anhand der Veränderunge im Zusammenhang mit der Steuerungsebene:

1. Trennung von Vermittlungs- und Steuerungsebene (*data* bzw. *control plane*)
2. Zentralisierung der Steuerungsebene
3. Programmierbarkeit der Steuerungsebene
4. Standardisierung der Schnittstellen

Die beiden Punkte, in denen die Beschreibungen übereinstimmen mögen SDN am stärksten charakterisieren:

- Externalisierung der Steuerungsebene aus den einzelnen Netzkomponenten
- Offenheit/Standardisierung der Schnittstellen zu der herausgetrennten Steuerungsebene

Die SDN-Architektur hat Vorfahren, die ihre wichtigsten Charakteristika beinhalten. Ein Beispiel ist der forCES-Ansatz [YDAG04], bei dem bereits eine Trennung von Vermittlung und Steuerung innerhalb eines Routerchassis vorgesehen ist, dargestellt in Abbildung 3.1. Ein solches Chassis kann mit mehreren Elementen (*Blades*) bestückt werden, davon mehrere Vermittlungselemente (*Forwarding Element, Router Blade*), die in das Chassis eingesteckt und die Vermittlungsentscheidung sowie die Ein-/Ausgabe auf das Medium beherrschen. Die Steuerungsebene ist in *Control Blades* getrennt umgesetzt. Die Verbindung zwischen den verschiedenen Elementen beider Ebenen geschieht über das interne Netz des Routerchassis (*Backplane*).

3.1.2 Komponenten

Die Trennung von Vermittlungs- und Steuerungsfunktionen widerspiegelt sich in den Gerätetypen: erstere werden von SDN-Geräten (engl. *SDN devices*) oder -Switches übernommen, letztere durch dedizierte Controller-Komponenten.

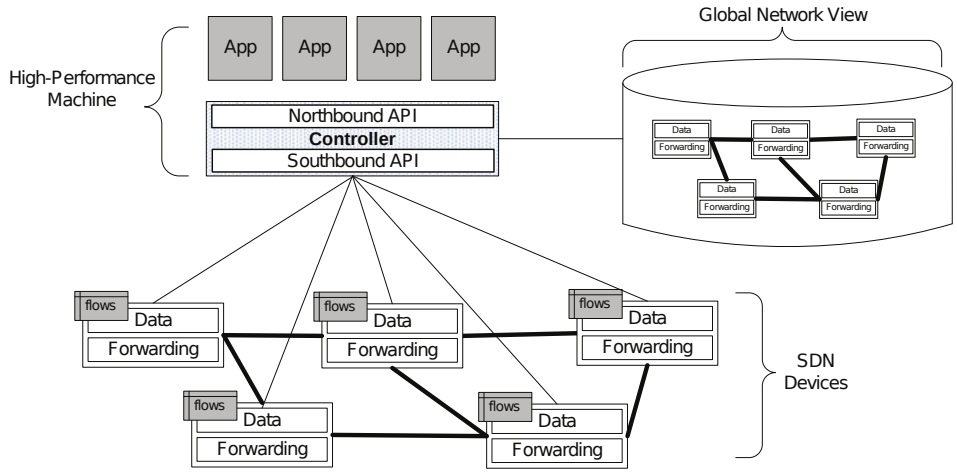

Abbildung 3.2: Generische Architektur für SDN (aus [GB14])

SDN-Geräte (Switches) SDN-Geräte sind "dumme Switches" mit zwei oder mehreren Netzschnittstellen (*ports*), die auf der Basis von Vermittlungstabellen (FIB) eingehende Nachrichten verarbeiten. Dabei können sie Protokollkontrollinformationen mehrerer Schichten (z.B. Schichten 2–4) berücksichtigen. Sie enthalten keinerlei Steuerungsfunktionen, sondern werden von dem Controller an einer (idealerweise standardisierten) Schnittstelle gesteuert.

Controller Der Controller ist die Steuerungskomponente für das Netz. Seine wichtigste Funktion ist die Bereitstellung eines Topologiedienstes, also einer topologischen Übersicht über das Netz. Hierzu erkennt erkennt Topologieänderungen (hinzukommende oder das Netz verlassende Leitungen, Endsysteme und SDN-Geräte). Der Controller verändert bei Bedarf die Konfiguration der SDN-Geräte.

Die vier grundlegenden Funktionen eines SDN-Controllers sind nach [GB14]:

- Entdeckung von Endgeräten
- Entdeckung von Netzkomponenten (SDN Devices)
- Topologiedienst
- Flow management: Verwaltung der konfigurierten Datenströme (siehe Abschnitt 3.2)

Weitere Funktionen des Netzes werden von SDN-Applikationen (siehe Abschnitt 3.1.4) realisiert.

3.1.3 Schnittstellen

Die Architektur beinhaltet also zwei Schnittstellen: jene zwischen SDN-Geräten und Controller und jene zwischen Controller und Applikationen. Entsprechend der Architekturskizze trennt

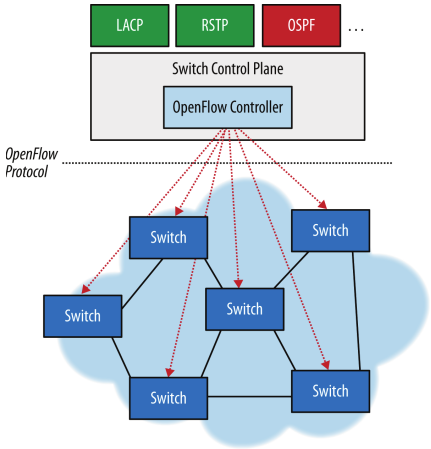

Abbildung 3.3: Anwendungsbeispiel der OpenFlow-Architektur (aus [NG13])

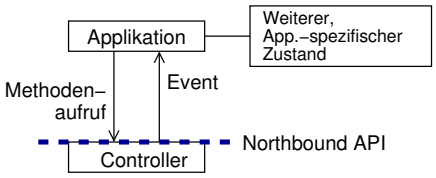

Abbildung 3.4: Interaktion zwischen SDN-Applikationen und Controller nach [GB14]

die *Northbound API* die Applikationen und den Controller (nördliche API, weil sie oben im Bild ist), die *Southbound API* ist die Schnittstelle zwischen Controller und SDN-Geräten.

Abbildung 3.3 zeigt ein Anwendungsbeispiel, bei der auf der Basis eines OpenFlow-Controllers Anwendungen, z.B. Routingverfahren wie OSPF, umgesetzt werden. Die Southbound API ist durch das OpenFlow-Protokoll realisiert.

3.1.4 SDN-Applikationen

Die Steuerungsfunktionen des Netzes (das Lernverhalten von Switches, Wegewahl, Sicherung der Dienstgüte, Prozeduren zur Datensicherheit, . . .) werden als Applikationen programmiert, die auf der Controller-Komponente laufen. Applikationen können die Konfiguration von Switches durch den Controller veranlassen, etwa zur Einführung einer Vermittlungsregel. Fällt zum Beispiel eine Leitung aus, kann eine Applikation für Wegewahl eine Umleitung des Netzverkehrs veranlassen, indem sie entsprechende Vermittlungsregeln in den Switches einfügen lässt. Unterschiedlich zu einem verteilten Wegewahlverfahren werden nicht auf allen betroffenen Knoten die Konfigurationsmaßnahmen berechnet.

Applikationen interagieren mit dem Controller an dessen oberen Schnittstelle (northbound API), wie in Abbildung 3.4 dargestellt.

Sie werden durch den Controller über Ereignisse im Netz benachrichtigt; sie sind in *Observer* bzw. *Listener* Rolle für abonnierte *Events*. Ereignisse umfassen Zustandsänderungen im Netz (Entdeckung neuer oder der Ausfall bekannter Netzkomponenten oder Endgeräten) oder das Eingehen bestimmter, von der Applikation abonnierten Nachrichten, denen entweder kein Flow (siehe Abschnitt 3.2) zugeordnet wurde oder für die die Aktion "Weiterleiten an Controller" in der Tabelle des empfangenden Switches angegeben wurde.[GB14] Dieses Verhalten bildet die Übergabe spezieller Typen von Nachrichten an die Steuerungsebene nach, z.B. die Nachrichten von Mangement- oder Routingprotokollen. Es bildet ebenfalls das Verhalten lernender Netzkomponenten nach (klassischen Switches oder Router, bei denen FPGA auf neue Verbindungen eingestellt werden).

3.2 Flows

SDN stützen sich auf das Konzept von *Flows*, das im folgenden umrissen wird.

Pfade bezeichnen Reihungen von Teilstrecken durch das Netz, die ein Paar von Endpunkten (DEE) verbinden; sie bilden eine Grundlage von IP-Wegewahl. Das Konzept eines Pfades ist rein topologisch: nur der Weg durch das Netz wird betrachtet, ohne Bezug auf die Typen und Inhalte des Datenverkehrs, der entlang des Pfades vermittelt wird.

Flows, also "Datenströme", sind gegeben durch die Menge der Nachrichten, die einem Ausdruck über ihre Attribute genügen. Attribute umfassen Adressen in der Protokollheadern der Nachricht wie auch Statusbits und andere Kennfelder; es kann auch der Ingressport (also der empfangende Port) der Netzkomponente betrachtet werden. Bei einem Flow betrachten wir somit auch die Typen (etwa das transportierte Protokoll) und u.U. auch die Inhalte des Datenverkehrs. Flows sind uni-direktional zwischen zwei Endpunkten im Netz. Ein Flow ist also gegeben durch:

- den Pfad
- die Belegung der Attribute
- die Richtung (als Spezialfall der Attributsbelegung von Adressen)

Beispiel: der Datenverkehr zwischen zwei Endpunkten A und B, die Audioströme und HTTP-Verkehr austauschen, kann also mittels verschiedener Flows charakterisiert werden:

1. der Pfad für Audiodaten von A nach B
2. der Pfad für Audiodaten von B nach A
3. der Pfad für HTTP von A nach B
4. der Pfad für HTTP von B nach A

Abbildung 3.5 zeigt auf der linken Seite, wie die vier Datenströme, die unter herkömmlicher, netzneutraler IP-Wegewahl entlang des gleichen Pfades vermittelt würden; die gleichen Datenströme (Flows) können aber auch auf getrennten Pfaden transportiert werden (rechte

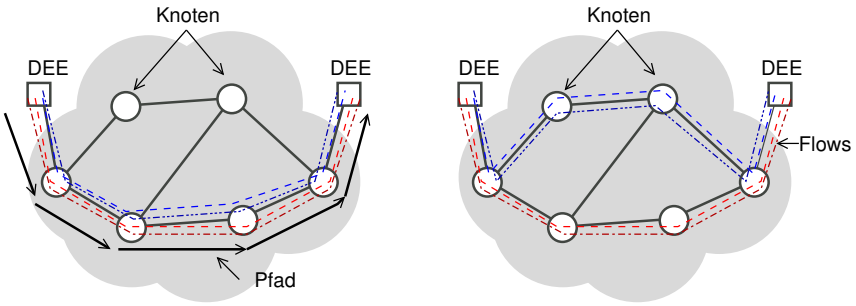

Abbildung 3.5: Flows entlang verschiedener Pfade

Seite). Zur Unterscheidung der Datenströme werden die Typen der transportierten Datenströme herangezogen. Diese Unterscheidung ermöglicht dem Netzbetreiber eine flexiblere Zuteilung von Netzkapazitäten an Dienste und Dienstnutzer. Gleichzeitig begründet sie die politische Diskussion um *Netzneutralität*.

Spezifikation von Flows Ein SDN-Gerät hält die Angaben zu Flows in Tabellen, den *flow tables*. Einträge der Tabellen beinhalten eine Musterangabe (*match fields*), die den Flow anhand der Protokollfelder spezifiziert sowie eine Aktion, die bei Erkennung eines Musters ausgeführt werden soll.

Die Werte der *match fields* können unterbelegt sein, d.h. manche Angaben können mit *wildcards* angegeben werden, um einen ganzen Wertebereich für das Muster anzugeben. Auf diese Weise kann die Anzahl der Einträge der Tabellen reduziert werden.

Die einem Muster zugeordnete Aktion ist aus dem Repertoire der SDN-Geräte: eine Nachricht kann auf einem bestimmten Port ausgegeben werden, sie kann verworfen werden, sie kann an den SDN-Controller (siehe 3.1.2) weitergeleitet werden, sie kann auf mehreren (oder allen) Ports ausgegeben werden (flooding, broadcast) oder sie kann modifiziert werden. Dieser Mechanismus ist vielleicht bekannt aus der Arbeitsweise von Paketfiltern wie etwa `iptables`.

Abbildung 3.6 zeigt die Primitive der Mustererkennung bei OpenFlow anhand seiner Tabelleneinträge: Ein Eintrag der Tabelle (eine Zeile) enthält zu einem Muster (Match Field) eine Aktion, die ausgeführt werden soll, wenn das Muster erkannt wird. Das Muster besteht aus einer Belegung der Protokollfelder in Schichten 2–4, die Aktion kann z.B. eine Weiterleitung extern/intern, eine Modifikation der Nachricht oder ihr Verwerfen sein.

Priorität Tabelleneinträge werden in Reihenfolge angewendet, worin eine Priorisierung der Einträge besteht. Spezifischere Angaben sollten also mit einer höheren Priorität versehen werden als allgemeinere. Eine Analogie findet sich im *longest prefix match*, der bei Routingtabellen angewendet wird: gibt es zu einer Zieladresse mehrere Einträge in der Routingtabelle, die zutreffen würden, wird der spezifischere (also jener mit dem längsten Prefix) angewendet.

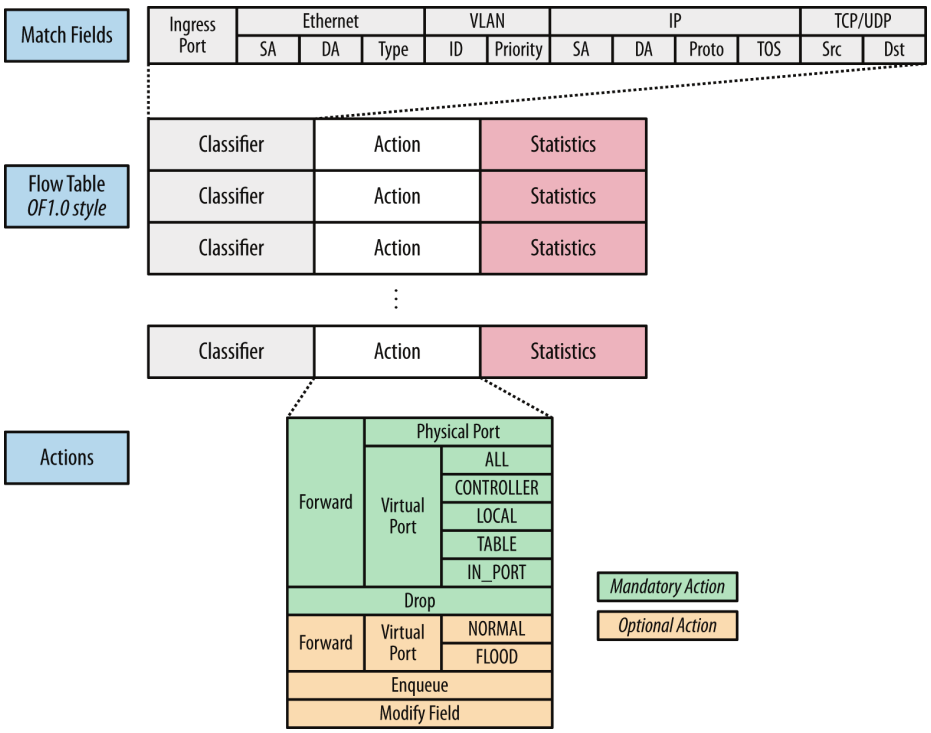

Abbildung 3.6: Primitive für die Mustererkennung bei OpenFlow 1.0 (aus [NG13])

Typen von Flows Flow-Spezifikationen können proaktiv oder reaktiv vorgenommen werden. Proaktive Flows werden von SDN-Applikationen (siehe Abschnitt 3.1.4) in ihrer Initialisierungsphase konfiguriert. Ein proaktiver Flow, der bis zur expliziten Änderung der Konfiguration besteht heisst *statisch*. Nicht-statische Flows können durch den Controller aufgrund von Ereignissen im Netz nachträglich verändert werden.

Reaktive Flows werden aufgrund von Entscheidungen außerhalb des Controllers (z.B. durch eine IDS-Applikation) spezifiziert.

4 Network Function Virtualization

Die bereits beschriebenen SDN können als eine neuartige Umsetzung der Netzsteuerung verstanden werden, wobei Steuerungsfunktionen aus den vermittelnden Netzkomponenten (Switches, Router) in eine Controller-Komponente zentralisiert werden.

Network Function Virtualization (*NFV*) ist ein ähnlicher Ansatz, der aber nicht speziell auf die Steuerungsebene des Netzes abzielt.

4.1 Idee

Die Funktionen eines Netzes beschränken sich nicht auf die reine Weiterleitung/Vermittlung von Daten und auf grundlegende Steuerungsfunktionen, sondern beinhalten auch Managementfunktionen jenseits der traditionell verteilt automatisierten Steuerung. Hierzu gehören Analysefunktionen (Deep Packet Inspection), Sicherheitsfunktionen, *Traffic Shaping*, etc (vgl. Abschnitt 2.1.7).

Problem Die Umsetzung solcher Netzfunktionen auf der Basis von spezialisierten, proprietären Hardwarekomponenten ("Middleboxes") ist beschränkend [CCea12]:

- Neue Netzfunktionen erfordern neue Hardwarekomponenten, die untergebracht, installiert und gemanagt werden müssen.
- Der Umgang mit ihnen (z.B. Konfiguration) erfordert Spezialwissen für jede Komponente.
- Hardware muss ausgetauscht werden, somit unterliegt die Netzfunktion an sich einem (unfreiwilligen) Wartungszyklus.

Prinzip Der Ansatz von NFV beschreibt die Umsetzung von Funktionen des Netzes außerhalb der vermittelnden Netzkomponenten und auch außerhalb von Middleboxes.

Das Prinzip von NFV kann also beschrieben werden als die Trennung von Funktion und (proprietärer) ausführender Plattform.[HSMA14]

4.1.1 Beispiel

Das Beispiel in Abbildung 4.1 zeigt die Virtualisierung einer Netzfunktion, *Network Function* (*NF*), ausgehend von einem traditionellen Netz bzw. eines SDN. Die beispielhaft betrachtete Netzfunktion ist ein *Intrusion Detection System* (*IDS*). Ein IDS analysiert den Netzverkehr und generiert Meldungen und Alarme, wenn Unregelmäßigkeiten im Netzverkehr erkannt werden.

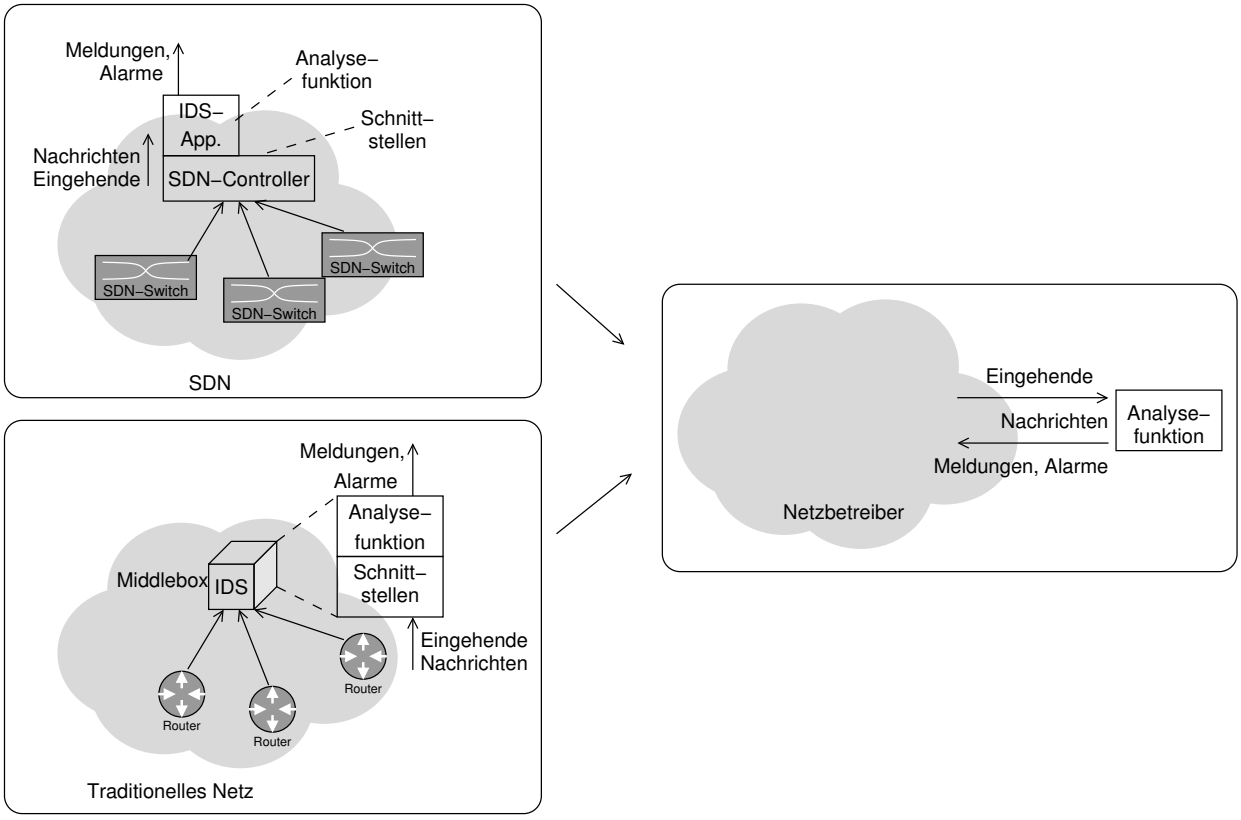

Abbildung 4.1: IDS als Middlebox, als SDN-Applikation und ausgelagert mittels NFV

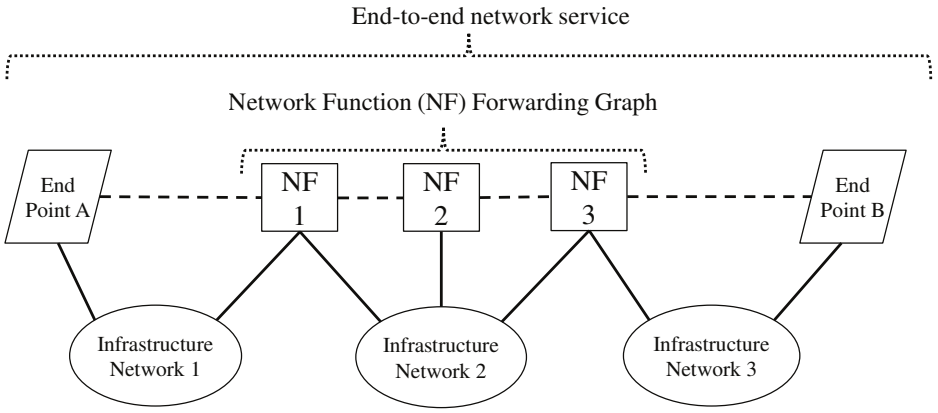

Abbildung 4.2: Ende-zu-Ende Dienst auf NFV-Basis (aus [rgs14])

Im ersten Fall (unten links im Bild) wird diese Funktion von einer Middlebox ausgeführt, im anderen Fall ist die NF als SDN-Applikation realisiert. Im NFV-Fall (rechts im Bild) wird die Funktion von der Netzhardware abstrahiert und auf einem Rechner ausgeführt, dem der relevante, zu analysierende Netzverkehr zugeführt wird und der die Analysefunktion ausführt. Das IDS wird somit als *Virtual Network Function* (*VNF*) umgesetzt.

4.2 Umsetzung

Die von ihrer speziellen Plattform getrennten Netzfunktionen sollen auf herkömmlichen Rechners oder aber auf der Basis von Infrastrukturdiensten anderer Anbieter ausgeführt werden.

Abbildung 4.2 zeigt einen Ende-zu-Ende Dienst, der aus mehreren im RZ ausgegeführten Netzfunktionen (*Virtual Network Function* (*VNF*)) zusammengestellt wird.

Die VNF können zu Gruppen zusammengefasst werden, die eine transparente Verkettung von Funktionen realisieren. Abbildung 4.3 illustriert einen solchen Fall.

4.2.1 Cloud Infrastrukturdienste

Infrastructure as a Service (*IaaS*) ist eine Klasse von Diensten, die unter den Terminus “Cloud” gefasst sind. Auf der Basis virtualisierter Systeme werden Dienstnehmern *Virtuelle Maschine* (*VM*) zur Verfügung gestellt, deren Nutzung abgerechnet wird. Zum Infrastrukturdienst gehören in der Regel auch Speicherelemente und Netzanbindung. Rechenkapazität kann auf diese Weise flexibel zu einer Dienstinstanz eines Kunden hinzugefügt (oder entfernt) werden.

Die bisher durch eine Middlebox erbrachten Funktionen könnten somit skalierbar und mandantenfähig auf der Basis eines Infrastrukturdienstes realisiert werden.

Auf der Basis einer virtualisierten Infrastruktur spezifiziert ETSI (European Telecommuni-

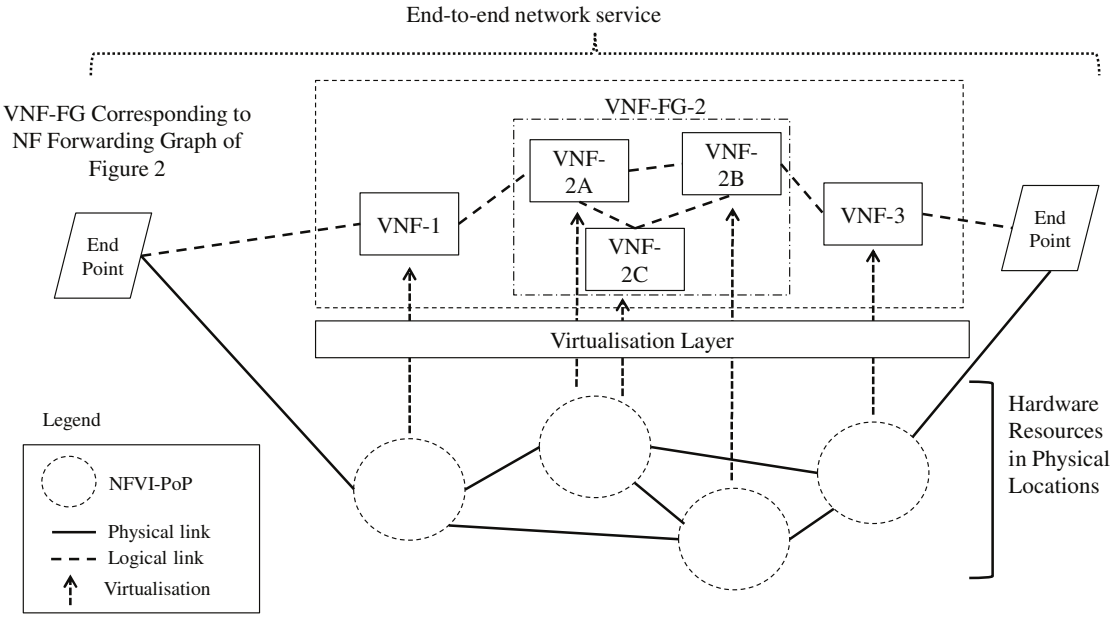

Abbildung 4.3: Funktionsgruppe als Spezialfall einer Netzfunktion (aus [rgs14])

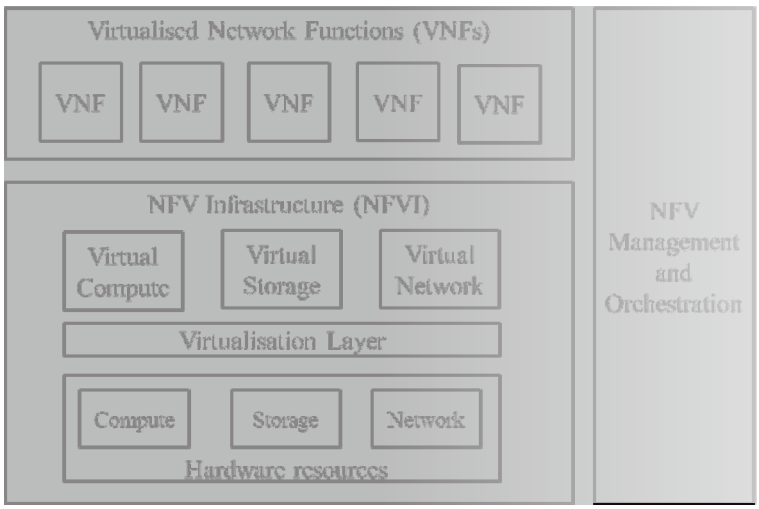

Abbildung 4.4: NFV Architekturrahmen (aus [rgs14])

cations Standards Institute) den Architekturrahmen in Abbildung 4.4: Netzfunktionen sind Nutzerinnen einer virtualisierten Infrastruktur und unterliegen der Orchestrierung (geordnete Zusammenstellung) und dem Management durch eine von der virtualisierten Infrastruktur gesonderten Verwaltungsfunktion (rechte Säule in Abbildung 4.4.

4.3 Gegenüberstellung SDN – NFV

Die beiden Themenbereiche SDN und NFV wirken ähnlich: in beiden Fällen werden Netzfunktionen aus den Hardwarekomponenten ausgegliedert und zentralisiert ausgeführt. Die folgende Aufstellung nach [HSMA14] vergleicht die beiden Ansätze:

Programmierbare Netze / SDN
- Entkopplung von Vermittlung und Steuerung
- Zentralisierung und Programmierbarkeit der Steuerung

Virtuelle Netzfunktionen / NFV
- Entkopplung von Netzfunktionen von spezialisierter Hardware (Appliance)
- Ausführung der Netzfunktionen auf generischer Hardware oder in virtuellen Maschinen

Zusammenarbeit von SDN und NFV
- SDN bietet programmierbare Verbindungen zwischen virtuellen Netzfunktionen
- NFV bietet einen Rahmen zur Implementierung zentralisierter SDN-Funktionen (z.B. SDN-Controller)

5 Seminarthemen

Die Infrastrukturschicht Wie funktionieren die “dummen Switches” der SDN an den Beispielen von OpenFlow und POF?

Netzbetriebssysteme und -controller Netzbetriebssysteme (NOS) verstehen sich als Sammlung von Verwaltungsmechanismen für Netze nach dem Vorbild von Betriebssystemen für Rechner. Welche Abstraktionen und Schnittstellen können für das Netz angeboten werden?

Network Hypervisors Virtualisierung von SDN: wie können mandantenfähige SDN-Partitionen erzeugt werden?

Programmiersprachen Ein “Software-defined” Ansatz erfordert eine Ausdrucksmöglichkeit für das Verhalten des Netzes. Welche Programmiersprachen wurden entwickelt, mit welchen Gemeinsamkeiten und Unterschieden?

Applikationen Wie und zu welchen Zwecken können die Eigenschaften von SDN genutzt werden?

Debugging, Testing und Verifikation Programmierbare Netze werden, wie der Name andeutet, von Programmen gesteuert, die einem Softwareentwicklungsprozess unterliegen. Wie können Tests, Fehlersuche und Verifikation umgesetzt werden?

Kopplung von SDN Wie können Verbundnetze aus SDN geschaltet werden?

SDN via existierender Schnittstellen Erlauben die existierenden Schnittstellen und Funktionen von Netzkomponenten eine Trennung der Kontroll- und Vermittlungsebenen?

Fallbeispiele für NFV Was bezweckt NFV und wie könnte es in verschiedenen Betriebsszenarien zum Tragen kommen?

NFV-Architektur Wie gliedern sich Netzfunktionen in die IT-Infrastruktur ein? (ETSI-Standards)

Standardisierung von SDN Welche Aspekte von SDN werden standardisiert, von wem, mit welchen Zielen? Welche Standardisierungsmaßnahmen fehlen?

Herausforderungen bei SDN Welche für SDN spezifischen Probleme/Herausforderungen gibt es, welche Lösungen hierfür wurden versucht?

Überblick der NFV-Standardardisierung Welche NFV-Aspekte werden standardisiert? Schwerpunkt auf die Standardisierung durch ETSI (European Telecommunications Standards Institute)

Herausforderungen bei NFV Welche Probleme/Herausforderungen ergeben sich durch die Nutzung von NFV (an Beispielen).

Netzneutralität: ein Opfer neuer Netze? Sowohl SDN als auch NFV bieten netzbasiert Funktionen zur attributsabhängigen Behandlung des Netzverkehrs an. Welche Auswirkungen kann das auf die Offenheit des Internet haben?

Abbildungsverzeichnis

Literaturverzeichnis

[CCea12] Margaret Chiosi, Don Clarke, and James Feger et. al. Network functions virtualisation – an introduction, benefits, enablers, challenges & call for action. White paper, European Telecommunication Standards Institute (ETSI), 2012.

[FASA+12] D. Fedyk, P. Ashwood-Smith, D. Allan, A. Bragg, and P. Unbehagen. IS-IS Extensions Supporting IEEE 802.1aq Shortest Path Bridging. RFC 6329 (Proposed Standard), April 2012.

[FLRW03] R. Frye, D. Levi, S. Routhier, and B. Wijnen. Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework. RFC 3584 (Best Current Practice), August 2003.

[GB14] Paul Göransson and Chuck Black. *Software Defined Networks – A Comprehensive Approach*. Morgan Kaufmann, 2014.

[Han06] M. Handley. Why the Internet only just works. *BT Technology Journal*, 24(3), July 2006.

[Hed88] C.L. Hedrick. Routing Information Protocol. RFC 1058 (Historic), June 1988. Updated by RFCs 1388, 1723.

[HSMA14] Hassan Hawilo, Abdallah Shami, Maysam Mirahmadi, and Rasool Asal. Nfv: State of the art, challenges and implementation. *IEEE Network Magazine*, November 2014. Preprint.

[IEE04] Media access control (mac) bridges. IEEE Standard for Local and metropolitan area networks 802.1d, IEEE Computer Society, June 2004.

[JP13] Raj Jain and Subharthi Paul. Network virtualization and software defined networking for cloud computing: A survey. *IEEE Communications Magazine*, November 2013.

[KREV+15] D. Kreutz, F.M.V. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76, January 2015.

[Moy98] J. Moy. OSPF Version 2. RFC 2328 (INTERNET STANDARD), April 1998. Updated by RFCs 5709, 6549, 6845, 6860, 7474.

[NG13] Thomas D. Nadeau and Ken Gray. *SDN: Software Defined Networks*. O'Reilly, 2013.

[Pos81] J. Postel. Internet Control Message Protocol. RFC 792 (INTERNET STANDARD), September 1981. Updated by RFCs 950, 4884, 6633, 6918.

[rgs14] Network functions virtualisation (nfv); architectural framework. Group Specification RGS/NFV-002, European Telecommunication Standards Institute (ETSI), December 2014.

[RLH06] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271 (Draft Standard), January 2006. Updated by RFCs 6286, 6608, 6793.

[Tan11] Andrew S. Tanenbaum. *Computer Networks*. Pearson Education, Prenctice Hall, 5 edition, 2011.

[TP09] J. Touch and R. Perlman. Transparent Interconnection of Lots of Links (TRILL): Problem and Applicability Statement. RFC 5556 (Informational), May 2009.

[YDAG04] L. Yang, R. Dantu, T. Anderson, and R. Gopal. Forwarding and Control Element Separation (ForCES) Framework. RFC 3746 (Informational), April 2004.