# Mjukvarudefinierade nätverk

## En introduktion

Daniel Eidenskog, Erik Hyllienmark och
Caroline Bildsten

FOI-R--5053--SE

December 2020

FOI

Daniel Eidenskog, Erik Hyllienmark och
Caroline Bildsten

# Mjukvarudefinierade nätverk

En introduktion

| | |
|---|---|
| Titel | Mjukvarudefinierade nätverk – En introduktion |
| Title | Software-defined networking – An introduction |
| Rapportnr/Report no | FOI-R--5053--SE |
| Månad/Month | December |
| Utgivningsår/Year | 2020 |
| Antal sidor/Pages | 67 |
| ISSN | 1650-1942 |
| Kund/Customer | Försvarsmakten |
| Forskningsområde | Informationssäkerhet |
| FoT-område | Operationer i cyberdomänen |
| Projektnr/Project no | E72877 |
| Godkänd av/Approved by | Christian Jönsson |
| Ansvarig avdelning | Ledningssystem |

Bild/Cover: Shutterstock, Pitiya Phinjongsakundit

# Sammanfattning

Mjukvarudefinierade nätverk ändrar modellen för hur trafikflöden styrs i nätverk genom att styrlogiken centraliseras och samtidigt separeras från nätverks-komponenternas datavägar. Tekniken syftar till att möjliggöra administrativt enklare styrning av trafikflöden, som dessutom kan ske på högre abstraktions-nivå. Mjukvarudefinierade nätverk är väl etablerade på marknaden, vilket innebär att flera populära kommersiella lösningar använder tekniken. Mjukvarudefinierade nätverk är vanliga i datacenter och trenden tycks vara att tekniken blir allt vanligare även i enklare systemlösningar.

Syftet med denna studie är att ge läsaren en grundläggande kunskap om mjukvarudefinierade nätverk och de byggstenar som används i dem. Kunskapen är främst avsedd att underlätta diskussioner kring mjukvarudefinierade nätverk i samband med utveckling och förvaltning av IT-system inom Försvarsmakten och andra offentliga organisationer.

Rapporten tar upp vad mjukvarudefinierade nätverk är, vilka byggstenar som krävs för att bygga sådana nätverk, hur de kan påverka nätverken ur olika aspekter och vilka effekter de kan få i IT-systemperspektivet. Rapporten har sin grund i vetenskaplig litteratur, som till största del centrerar kring öppna lösningar för mjukvarudefinierade nätverk, då information saknas i djupet om proprietära lösningar.

Många av de problem som hanteras med hjälp av mjukvarudefinierade nätverk kan även lösas med andra tekniker. Mjukvarudefinierade nätverk för dock även med sig andra fördelar, exempelvis underlättandet av administrationen av nätverket. Dessutom förbättrar tekniken möjligheterna för nyttjandet av andra lösningar så som tjänstekedjor eller mikrosegmentering. Mjukvarudefinierade nätverk för dock inte endast med sig fördelar utan introducerar även nya angreppsytor och fallgropar, såsom att den centrala styrlogiken blir ett attraktivt mål för angripare.

Nyckelord: mjukvarudefinierade nätverk, nätverksvirtualisering, virtuella nätverksfunktioner, tjänstekedjor, SDN, NFV, VNF.

# Summary

Software-defined networking alters the model for traffic flow control in computer networks by centralizing the control logic while also separating it from the data paths in the network components. This enables simpler administration of traffic flows, facilitating control at a higher abstraction level. Software defined networking is well established in the market, which shows in that it is employed by several popular products for highly complex IT systems. Software-defined networking has a firm foothold in data centers, with a trend towards also being used for IT systems with lower complexity.

The purpose of the study is to provide the reader with a base-line knowledge on software-defined networking and the building blocks used. This knowledge is intended to facilitate discussions on software-defined networking when developing and maintaining IT systems, primarily within the Swedish Armed Forces and other public organizations.

The report introduces what software-defined networking is and which building blocks are used. The report also discusses how networks can be affected by software-defined networking and how it relates to the IT-system perspective. The report is based on scientific publication, which largely centers on open solutions of software-defined networks, as there is a lack of in-depth information about proprietary solutions.

Even though many of the problems that software-defined networking handles could be solved with previously available techniques, software-defined networking has proven itself as a useful technique that can simplify network administration, while facilitating the use of other techniques such as service chaining and micro segmentation. In addition to the benefits from software-defined networking, there are also drawbacks such as new potential pitfalls and attack surfaces, such as the central controller becoming an attractive target for adversaries.

Keywords: software-defined networking, network virtualization, virtual network functions, service chains, SDN, NFV, VNF.

# Innehållsförteckning

# 1 Inledning

Datornätverk tenderar att bli komplexa och svåradministrerade, särskilt i större IT-system. Många olika typer av utrustning och funktioner samsas i heterogena nätverk som transporterar många sorters trafik med olika funktionella behov. Denna komplexitet ökar arbetsbördan för nätverksadministratörer och ökar även risken för felkonfigurationer eller misstag, vilket kan leda till allehanda problem i IT-systemen. Kostsam administration, brist på flexibilitet och svårupptäckta fel som ger kaskadeffekter är bara några exempel på följder som kan uppstå på grund av komplexiteten.

Mjukvarudefinierade nätverk (eng. software-defined networking, SDN) hanterar delar av komplexiteten i datornätverk genom att centralisera hur nätverket styr trafikflöden. Genom denna centralisering kan nätverkets beteende programmeras centralt med hjälp av olika nätverksapplikationer, vilket bidrar till flexibilitet och förenklar möjligheten till att anpassa nätverkets beteende utifrån det aktuella behovet. Idéerna och teknikerna bakom mjukvarudefinierade nätverk är inte nya utan har utvecklats sedan 1990-talet fram till de lösningar som finns idag.

Den mest fundamentala skillnaden mellan traditionella nätverk och mjukvarudefinierade nätverk är att styrningen av trafiken – det vill säga formuleringen av regler för beslut på paketnivå om vart trafiken ska skickas vidare – inte längre utförs av de funktioner som vidareförmedlar paketen. Styrningen samlas i stället i en eller flera samverkande styrenheter som beslutar om trafikflöden på övergripande nätverksnivå.

Att något är mjukvarudefinierat (eng. software-defined) har varit lite av ett modebegrepp under ett antal år, såsom i mjukvarudefinierade *nätverk,* mjukvarudefinierad *lagring*, mjukvarudefinierade *beräkningar* och mjukvarudefinierade *datacenter*. Mjukvarudefiniering betyder lite olika saker beroende på i vilket av dessa sammanhang det används, men i IT-sammanhang handlar det generellt sett, om separation av styrning och administration från de underliggande funktioner som bygger upp nätverken, lagringssystemet, beräkningsnoderna eller datacentren.

Mjukvarudefinierade nätverk har skapat möjligheter till nya lösningar för hur nätverk konstrueras och hanteras. Exempelvis har tekniken lett till nya idéer rörande segmentering, lastbalansering, tjänstekedjor, färdvägsberäkningar och säkerhet. Idag används mjukvarudefinierade nätverk bland annat i datacenter och förväntas spela en stor roll i nya mobilnätverk. De är dock ännu relativt ovanliga i exempelvis kontorsnätverk.

Mjukvarudefinierade nätverk medför dock inte bara fördelar utan introducerar också nya fallgropar och angreppsytor. Exempelvis kan bristande interoperabilitet mellan komponenter från olika leverantörer leda till

inlåsningseffekter eller att flexibiliteten begränsas. Dessutom kan ogenomtänkta implementationer ge problem med prestanda och skalbarhet. Utöver detta tillkommer säkerhetsmässiga aspekter, såsom att styrenheterna blir en attraktiv angreppspunkt, vilket ställer krav på genomtänkta skyddsåtgärder.

I den här rapporten kommer läsaren att få fördjupa sig i mjukvarudefinierade nätverk samt de byggstenar i form av nätverksfunktioner och protokoll som används. Rapporten centrerar kring öppna lösningar för mjukvarudefinierade nätverk, då information saknas i djupet om de proprietära. Mjukvarudefinierade nätverk sätts sedan in i ett systemperspektiv som belyser några viktiga skillnader mot traditionella nätverk. Slutligen diskuteras mjukvarudefinierade nätverk utifrån tre olika perspektiv – tekniken, säkerheten och Försvarsmakten.

## 1.1 Syfte och mål

Syftet med studien är att ge läsaren en grundläggande kunskap om mjukvarudefinierade nätverk och de byggstenar som används i dessa typer av nätverk. Kunskapen är avsedd att underlätta diskussioner kring mjukvarudefinierade nätverk i samband med utveckling och förvaltning av IT-system i Försvarsmakten.

Målet med studien är att beskriva vad mjukvarudefinierade nätverk är och vilka egenskaper de innehar utifrån de byggstenar som används. För att nå detta mål, fokuserar studien på följande frågeställningar:

- Vad är mjukvarudefinierade nätverk?
- Vilka byggstenar använder mjukvarudefinierade nätverk?
- Vilka är skillnaderna mellan mjukvarudefinierade nätverk och traditionella nätverk?

Studien har utförts inom ramen för Försvarsmaktens samlingsbeställning inom forskning och teknikutveckling (FoT). Studien har genomförts i projektet *IT-säkerhetsmetoder* som ingår i FoT-området *Operationer i cyberdomänen*. Rapporten riktar sig huvudsakligen till personer som arbetar med anskaffning, utveckling och förvaltning av Försvarsmaktens IT-system.

## 1.2 Läsanvisning

Kapitel 2 ger en bakgrund till datornätverk och historiken bakom mjukvarudefinierade nätverk.

Kapitel 3 beskriver de olika planen i mjukvarudefinierade nätverk samt teknikens olika egenskaper.

Kapitel 4 beskriver mer detaljerat vad nätverksfunktioner är och hur de fungerar.

Kapitel 5 beskriver några viktiga protokoll som används i mjukvarudefinierade nätverk.

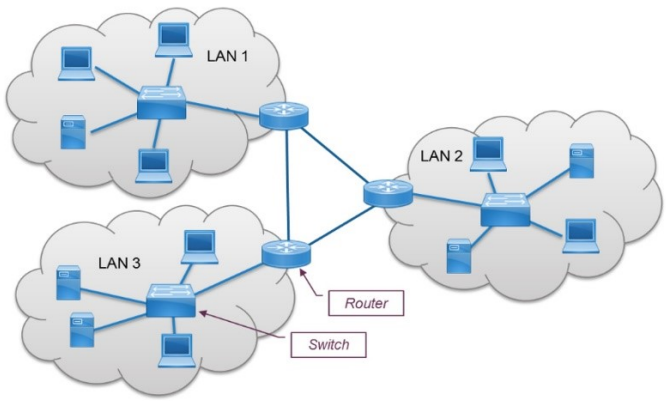
Kapitel 6 problematiserar kring mjukvarudefinierade nätverk ur ett systemperspektiv.

Kapitel 7 innehåller diskussion och slutsatser.

# 2 Datornätverk

Datornätverk är grunden för kommunikation mellan olika datorer. Nätverken finns i många olika varianter med olika egenskaper, men bygger i mångt och mycket på samma eller liknande tekniker.

I botten finns den fysiska överföringen av information, exempelvis i form av elektriska signaler i fysiska kablar, ljuspulser i optiska fibrer eller modulerade radiovågor.[1] Den fysiska överföringen sker mellan olika enheter som kan vara *nätverksfunktioner*, ofta någon form av brygga mellan två fysiska länkar, eller *ändutrustning*, i form av exempelvis datorer.

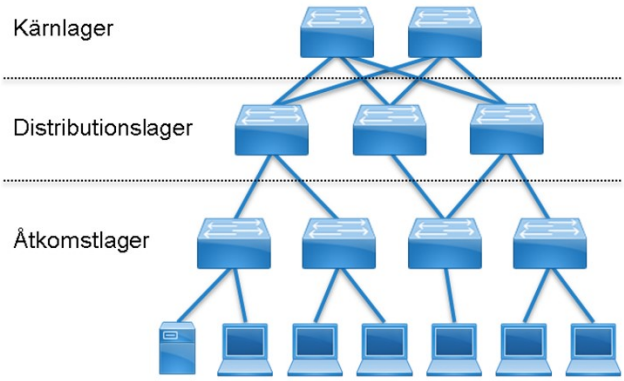



Figur 1. Sammankopplade lokala nätverk.

Nätverksfunktionerna är de fysiska eller virtuella enheter i nätverket som huvudsakligen bestämmer nätverkets funktion och beteende. De enklaste formerna av nätverksfunktioner används för att förmedla trafik mellan andra entiteter i nätverket. De två viktigaste typerna av nätverksfunktioner är *switchar* och *routrar*. Förenklat går det att säga att en switch sammanbinder enheter i ett nätverk medan routrar sammanbinder flera nätverk.[2] Detta illustreras i figur 1 som visar tre lokala nätverk (eng. local area network, LAN), vart och ett uppbyggt kring en enda switch. De tre nätverken är sammanlänkade genom att routrarna för respektive nätverk kopplats till varandra.

[1] Det finns stor frihet i vilka fysiska tekniker som går att använda i nätverken. Exempelvis finns det en standard för hur brevduvor kan vara den fysiska bäraren i nätverket, se https://tools.ietf.org/html/rfc1149 [läst 2020-04-28]. Standarden togs visserligen fram som ett aprilskämt men har testats och visat sig fungera i verkligheten (även om brevduvor inte är en särskilt praktisk lösning för kommunikation), se https://www.blug.linux.no/project/rfc1149/ [läst 2020-04-28].

[2] I praktiken är begreppsanvändningen betydligt mer komplicerad. Det finns olika typer av switchar och routrar som i princip utgör en glidande skala. I dag finns det betydande överlapp i de funktioner som återfinns i olika produkter.




Figur 2. Ciscos trelagersmodell för nätverk.

Större företagsnätverk är betydligt mer komplexa i sin uppbyggnad än exemplet i figur 1. Dessa kan i stället byggas upp enligt en hierarkisk struktur som exempelvis organiseras enligt Ciscos trelagersmodell, se figur 2 (Cisco, 2014a). Ciscos modell bygger på tre lager – *kärnlagret*, *distributionslagret* och *åtkomstlagret*.[3] Åtkomstlagret består av de switchar som ligger närmast användarna, där en hög andel av switcharnas portar typiskt används för att ansluta till användarnas utrustning. Distributionslagret aggregerar trafiken från åtkomstlagret till kärnlagret, samtidigt som trafiken exempelvis filtreras utifrån policyer. Kärnlagret utgör nätverkets ryggrad i form av switchar och routrar som tillsammans bör ge tillförlitlighet, skalbarhet och hög prestanda.

I praktiken är dock nätverk i regel betydligt mer komplexa än i figur 2. De är ofta betydligt mer heterogena och mindre strukturerade, samtidigt som de består av många olika typer av nätverksfunktioner med olika egenskaper. En vanlig förändring är exempelvis att kärn- och distributionslagren slås samman i ett kombinerat lager bestående av mer komplexa nätverksfunktioner (Pueblas m.fl., 2010).

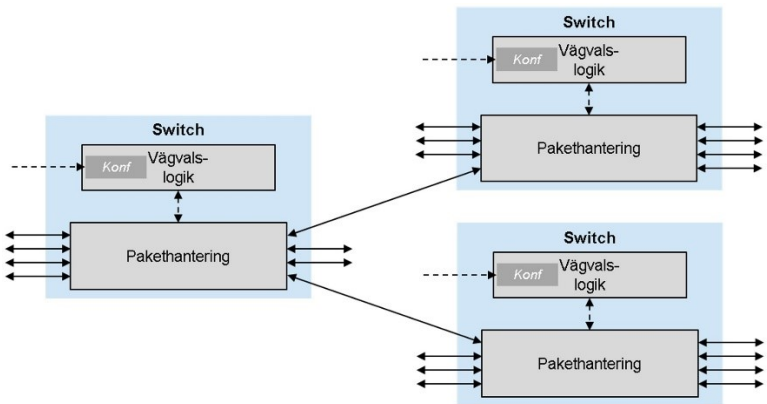
De enklaste switcharna, på engelska kallade *unmanaged switches*, kräver inga inställningar utan kan kopplas in i nätverket och utföra sin uppgift direkt. I princip alla andra nätverksfunktioner kräver att mer eller mindre omfattande inställningar – så kallad *konfiguration* – görs för att funktionen ska kunna utföra sina uppgifter. Konfigurering av nätverkets funktioner kan bli ett omfattande arbete, speciellt om det görs manuellt i mer komplexa nätverk. Därför finns ett antal olika verktyg för att skapa, hantera och distribuera konfigurationer. Konfigurationsverktygen har ofta en bredare funktion så att de även kan konfigurera andra delar av IT-systemet, såsom servrar, systemtjänster och

[3] De tre lagren benämns *core layer*, *distribution layer* och *access layer* på engelska (Cisco, 2014a).

användarutrustning. Exempel på sådana verktyg är Puppet[4], Ansible[5] och Saltstack[6].

Många nätverksfunktioner bygger på samma fundamentala grundprincip: först undersöks det inkommande paketet, därefter tas någon form av beslut om vad nätverksfunktionen ska göra med paketet och sist utförs det fattade beslutet. Hos switchar och routrar innefattar beslut typiskt vägvalet för paketets fortsatta resa genom nätverket. För brandväggar gäller beslut istället om att kasta eller vidareförmedla paketet.

Traditionella nätverk exemplifieras av de tre sammankopplade switcharna i figur 3. Nätverksfunktionerna – i detta fall switchar – tar sina egna beslut om vilken port som respektive paket ska vidarebefordras på utifrån en vägvalslogik eller ett regelverk som byggts in av tillverkaren och som i varierande grad kan konfigureras av systemadministratörer. Konfigurationen är typiskt något relativt statiskt som bara ändras vid behov, till exempel när nätverket byggs om. I *mjukvarudefinierade nätverk* centraliseras vägvalslogiken till en eller flera styrenheter, som sedan delegerar utförandet av beslut till nätverksfunktioner. Detta illustreras av exemplet i figur 4, där nätverksstrukturen återigen byggs upp av tre switchar. Dessa saknar egen vägvalslogik och förlitar sig i stället på en central styrenhet för besluten om vilka vägar paketen ska ta.

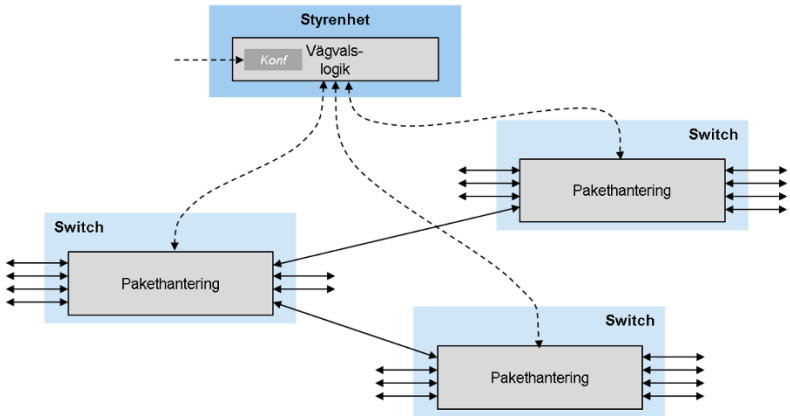



Figur 3. Tre sammankopplade, traditionella switchar med inbyggd vägvalslogik.

[4] http://www.puppet.com

[5] http://ansible.com

[6] Saltstack (http://saltstack.com) köptes upp av VMware i oktober 2020, se https://blogs.vmware.com/management/2020/10/vmware-completes-saltstack-acquisition-to-bolster-software-configuration-management-and-infrastructure-automation.html [läst 2020-11-10].




Figur 4. Mjukvarudefinierat nätverk, där vägvalslogiken har centraliserats till en styrenhet.

En styrenhet som ensamt sköter nätverket och som råkar ut för driftstörningar skulle innebära tillgänglighetsproblem i nätverket. I praktiken sköts styrningen i stället av flera samverkande och synkroniserade styrenheter för att åstadkomma redundans i systemet.

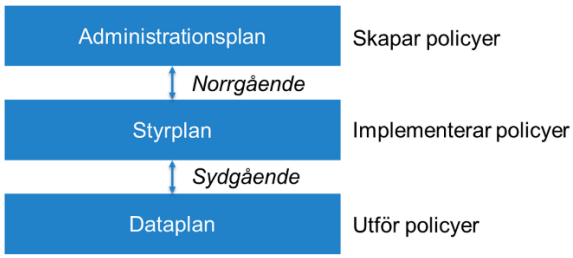
Det distribuerade beslutsfattandet – det vill säga att styrenheterna tar beslut som nätverksfunktionerna effektuerar – sker normalt sett inte på paket-för-paket-basis då det skulle ge problem med exempelvis överdrivna fördröjningar och överlast på styrenheterna. I stället sker styrningen baserat på trafikflöden. Styrenheterna har en bild över nätverket – ibland kallad för en *global vy* över nätverket – som exempelvis visar vilka nätverksfunktioner som finns och hur de är sammankopplade samt vilka trafikflöden som är aktiva. Den globala vyn används sedan som utgångspunkt när beslut tas för nya eller förändrade trafikflöden.

## 2.1 Nätverkens tre funktionsplan

Funktioner i datornätverk kan delas in i tre funktionsplan som tillsammans täcker in de olika aspekterna av att vidareförmedla trafik i nätverket. De tre planen är (Kreutz m.fl., 2015):

- *Administrationsplanet* som skapar nätverkspolicyer.
- *Styrplanet* som implementerar nätverkspolicyer.
- *Dataplanet* som utför nätverkspolicyer.

Nätverkspolicyerna beskriver olika regler som ska tillämpas på trafiken i nätverket. Policyerna kan exempelvis beskriva vägval för olika trafiktyper, hur trafiken i nätverket ska fördelas och hur nätverket ska upprätthålla garantier på tjänstekvalitet för vissa trafikflöden.




Figur 5. Nätverkens tre funktionsplan.

Figur 5 visar de tre funktionsplanen och hur de relaterar till varandra. Gränsytan mellan styrplanet och administrationsplanet kallas typiskt för det *norra* eller *norrgående* (eng. northbound) gränssnittet. På motsvarande sätt kallas gränsytan mellan styrplanet och dataplanet för det *södra* eller *sydgående* (eng. southbound) gränssnittet. Trafik inom respektive plan, exempelvis mellan två styrenheter, kallas ofta för *öst-västlig*. I denna rapport används termen öst-västlig uteslutande för trafik inom styrplanet.

Administrationsplanets ansvar är att skapa nätverkspolicyer, exempelvis gällande routing, brandväggsregler, lastbalansering och nätverksövervakning.

Styrplanet implementerar administrationsplanets nätverkspolicyer genom att skapa motsvarande regler, exempelvis i form av flödestabeller, som sedan distribueras till dataplanet. Reglerna bygger på policyerna som kompletteras med statisk och dynamisk information om nätverket och de enheter som kopplats till nätverket. Delar av denna information kommer från dataplanet.

Dataplanet står för inspektion och vidareförmedling av paket i nätverk, där förmedlingsbesluten baseras på de regler som publicerats av styrplanet. När dataplanet saknar information om hur ett paket ska hanteras behöver styrplanet informeras så det i sin tur kan publicera uppdaterade regler, exempelvis genom kompletterade flödestabeller.

Indelningen i tre funktionsplan görs utmed en annan dimension än Ciscos trelagersmodell som togs upp tidigare i kapitlet. Cisco-modellen delar in nätverksfunktionerna efter deras roller i nätverket, medan funktionsplanen delar in nätverket utifrån hur trafikstyrlogiken fördelas mellan olika entiteter. Som exempel på detta kan en enklare, fristående switch nära användarna därmed återfinnas på åtkomstlagret i Cisco-modellen, medan den implementerar både styrplan och dataplan.

Mjukvarudefinierade nätverk innebär i sin mest grundläggande form att styrplanet – som inkluderar vägvalslogiken – flyttas ut från nätverksfunktionerna för att i stället centraliseras till en eller flera styrenheter som är gemensamma för alla nätverksfunktioner. Dataplanet – det vill säga själva paketförmedlingen – lämnas kvar i nätverksfunktionerna även i mjukvarudefinierade nätverk.

Administrationsplanet är inte så väldefinierat i vare sig traditionella nätverk eller mjukvarudefinierade nätverk. I traditionella nätverk består det administrativa planet typiskt av att nätverksadministratörerna konfigurerar nätverksfunktionerna utifrån de olika policyer som ska gälla i nätverket. I mjukvarudefinierade nätverk kan det administrativa planet antingen bestå av manuellt angivna policyer eller av applikationer som skapar nätverkspolicyer, exempelvis utifrån ett specifikt beteende som önskas i nätverket eller utifrån en policy på systemnivå.

Separationen av styr- och dataplan är essensen i mjukvarudefinierade nätverk. I ett mjukvarudefinierat nätverk ges det centraliserade styrplanet ett helhetsansvar för hur nätverkspolicyerna implementeras för hela nätverket, samtidigt som mycket av ansvaret lyfts bort från nätverksfunktionerna.

## 2.2 Terminologi

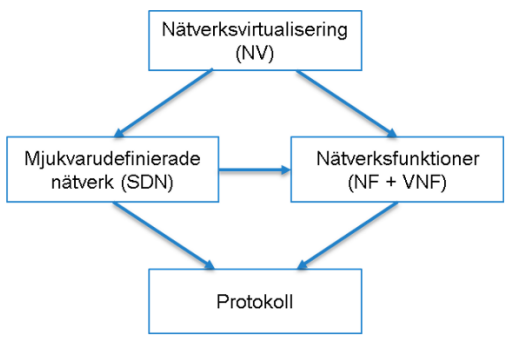
Utöver de tre funktionsplanen som togs upp i föregående avsnitt förekommer ett antal olika begrepp i samband med mjukvarudefinierade nätverk. Dessa olika begrepp motsvarar olika tekniker och byggstenar som typiskt används i moderna datornätverk och IT-system.

Nedan beskrivs de viktigaste begreppen i korthet för att etablera kontexten för rapportens fortsatta innehåll. Samtliga begrepp kommer att behandlas mer utförligt senare i rapporten.

- *Mjukvarudefinierade nätverk* (eng. software-defined networking, SDN) är ett samlingsnamn för tekniker som centraliserar nätverkets styrplan till en styrenhet eller flera samverkande styrenheter.
- *Trafikflöden* (eng. traffic flows) är en samling paket som skickas mellan samma sändare och mottagare, på samma portar och med samma protokoll inom en sammanhängande tidsperiod.
- *Nätverkspolicyer* (eng. network policy) är regeluppsättningar som beskriver hur nätverken ska bete sig, exempelvis hur trafik ska ledas genom nätverket. Nätverkspolicyerna kan beskrivas på olika abstraktionsnivåer, från övergripande beteende på nätverksnivå till detaljer om exempelvis enskilda trafikflöden.
- *Nätverksvirtualisering* (eng. network virtualization) är tekniker för att skapa logiska nätverk som är frikopplade från det underliggande fysiska nätverket. Samma fysiska nätverk kan bära flera virtuella nätverk som delar på det fysiska nätverkets resurser.
- *Nätverksfunktioner* (eng. network functions, NF) är funktioner vars huvudsyfte är att operera på trafiken i nätverket. Därmed arbetar nätverksfunktionerna alltid i dataplanet, men kan beroende på utförande även inkludera hela eller delar av styrplanet. Nätverksfunktionerna utgörs av förmedlingsfunktioner och nätverkssäkerhetsfunktioner.

- *Förmedlingsfunktioner* (eng. forwarding functions) är nätverksfunktioner vars huvuduppgift är att förmedla trafik, såsom switchar och routrar.
- *Nätverkssäkerhetsfunktioner* (eng. network security functions) är nätverksfunktioner vars huvuduppgift är säkerhetsfokuserad, exempelvis brandväggar och intrångsdetektorer (IDS).
- *Virtualisering av nätverksfunktioner* (eng. network function virtualization, NFV) är ett koncept som innebär att nätverksfunktioner implementeras i en virtuell miljö i stället för i fysiska nätverksenheter. Begreppet NFV brukar ibland användas om såväl konceptet som de virtuella nätverksfunktionerna.
- *Virtuella nätverksfunktioner* (eng. virtual network function, VNF) är nätverksfunktioner som implementerats i en virtuell miljö.
- *Protokoll* utgör de överenskommelser som definierar hur olika utrustningar och funktioner kommunicerar. Protokoll är typiskt standardiserade regelverk för hur meddelanden ska utformas, hanteras och tolkas.
- *Virtualisering* är ett samlingsnamn för tekniker som skapar en virtuell miljö för ett IT-system eller en funktion i ett IT-system. Den virtuella miljön efterliknar typiskt en fysisk miljö, såsom en server.

Figur 4 visar inbördes beroenden mellan några av rapportens centrala begrepp. Nätverksvirtualisering kräver att det finns ett underliggande nätverk och därmed krävs nätverksfunktionerna, oavsett om de är fysiska eller virtuella. Nätverksvirtualisering kräver däremot inte att det finns virtuella maskiner i systemet, exempelvis i form av virtualiserade nätverksfunktioner eller virtualiserade servrar.




Figur 6. Inbördes beroenden mellan några av de centrala begreppen.

Mjukvarudefinierade nätverk är inte heller ett krav för nätverksvirtualisering då det redan tidigare funnits tekniker för att hantera virtuella nätverk.[7] Mjukvarudefinierade nätverk kan däremot ses som ett alternativt sätt att skapa och hantera virtuella nätverk, där dessa definieras genom nätverkspolicyer som skapas i applikationsplanet och implementeras på styrplanet.

Nätverksfunktioner – fysiska och virtuella – används för att bygga upp nätverket och är därmed ett krav för att kunna genomföra nätverksvirtualisering eller bygga ett mjukvarudefinierat nätverk. Såväl mjukvarudefinierade nätverk som nätverksfunktioner kräver väldefinierade protokoll för kommunikationen.

## 2.3 Framväxten av mjukvarudefinierade nätverk

Sättet att bygga och konfigurera datornätverk har utvecklats i takt med att datorsystemen har utvecklats, vilket bland annat innebär att nya paradigm har tillkommit över årens lopp. Mjukvarudefinierade nätverk utgör ett sådant nytt paradigm, vars historik kortfattat presenteras i detta avsnitt. Historiken utgår från Feamsters m.fl. (2014).

### 2.3.1 Aktiva nätverk

När användningen av internet ökade kraftigt under 1990-talet skapades många nya applikationer och användningsområden för det nya, globala nätverket. Den långsamma processen för att standardisera nya protokoll och nya nätverkstjänster ledde till viss frustration i forskarvärlden, varvid ett nytt forskningsområde öppnade sig. Drivkraften var att skapa mer flexibla nätverk där det lättare skulle gå att införa nya protokoll och nya principer för att styra nätverk. Med nätverksfunktioner som i någon grad gick att programmera utifrån, ansåg vissa forskare att funktionerna skulle kunna bli betydligt mer anpassningsbara. Genom anpassningsbarheten skulle nya protokoll och principer kunna införas i systemen utan att behöva vänta på långsamma och frustrerande standardiseringsprocesser samt att tillverkarna inför funktionerna i sina produkter.

Forskningsinitiativen resulterade bland annat i det som kom att kallas *aktiva nätverk* (eng. active networking). I aktiva nätverk införs gränssnitt som gör nätverksfunktionernas resurser externt programmerbara, vilket innebär att en separat styrenhet till viss del kan bestämma beteendet hos nätverksfunktionerna. Detta medför bland annat att nätverksfunktionerna kan utföra externt

[7] Exempelvis kan systemen bygga på *Multiple Registration Protocol* (MRP) (IEEE, 2018) eller Ciscos *VLAN trunking protocol* (VTP). MRP är en vidareutveckling av *GARP* VLAN Registration Protocol* (GVRP) som definierades år 1998 (IEEE, 1999), det vill säga långt innan framväxten av mjukvarudefinierade nätverk. *Generic Attribute Registration Protocol.

programstyrda operationer på utvalda delmängder av alla paket som passerar dessa programmerbara funktioner.

Mycket av forskningen på aktiva nätverk finansierades av den amerikanska försvarsforskningsorganisationen DARPA[8]. Arbetet med aktiva nätverk utmärktes av två centrala tankar:

- Att göra nätverkskomponenter programmerbara.
- Att öppna upp nätverkskomponenter från olika leverantörer genom standardiserade gränssnitt.

Utvecklingen av aktiva nätverk fokuserade på de enskilda nätverksfunktionerna. Aktiva nätverk användes dock inte i någon större utsträckning, bland annat för att det saknades en tydlig drivkraft och för att det krävde anpassningar av nätverksfunktionerna för att fungera.

### 2.3.2 Separera styr- och dataplan

Stigande trafikmängder och ökande behov av tillförlitlighet, förutsägbarhet och prestanda i nätverken efter år 2000 gjorde att nätverksoperatörerna behövde mer praktiska verktyg för att exempelvis styra trafik. Ett av de problem som uppdagades var den täta kopplingen mellan styr- och dataplanet i routrar och switchar, vilket gjorde det svårt att felsöka konfigurationsproblem samt att förutse och styra nätverkens beteende.

Vid denna tid ökade även överföringshastigheten på länkarna i internetoperatörernas stamnätverk. Detta gjorde att internetoperatörerna fick kämpa med att hantera sina växande nätverk för att upprätthålla tillförlitligheten och samtidigt skapa nya tjänster åt sina kunder. Hastighetsökningen byggde delvis på att leverantörerna av nätverksutrustning hade börjat implementera datavägarna för vidarebefordring av paket direkt i dedikerad hårdvara, så kallade ASIC[9]-chip, vilket på sätt och vis kan ses som ett första steg i en separation av styr- och dataplan. ASIC-chippen tog hand om logiskt enkla aspekter av kommunikationen (såsom själva vidareförmedlingen av paketen) med hög prestanda medan mer komplexa beslut (såsom routing-beslut) togs av en långsammare processor (Singhal & Jain, 2002). Tillsammans verkade dessa faktorer för en tydligare separation mellan styr- och dataplan. Med utvecklingen av nya, öppna protokoll som Forces[10] gick det att flytta styrplanet från nätverkskomponenterna till en eller flera centraliserade servrar. När det är flera styrenheter som sköter styrningen av nätverket måste dessa samverka på ett bra sätt. Under perioden

[8] Defense Advanced Research Projects Agency

[9] Application-specific Integrated Circuit

[10] Protokollet *Forwarding and Control Element Separation* (ForCES) publicerades i ett första utkast 2004. Mer information om protokollet återfinns i avsnitt 5.2.

forskades det därför bland annat om hur otillgängliga styrenheter och icke-konsistenta nätverkstillstånd kan hanteras.

Några av de saker som utmärker denna period är:

- Centralisering av den logiska styrningen av nätverk.
- Framväxt av öppna gränssnitt och protokoll mellan styr- och dataplan.
- Nya möjligheter för visualisering av hela nätverk.

Även om de protokoll och tekniker som kom fram under perioden innebar ett ganska stort steg på den tekniska sidan fick de inget omfattande genomslag, delvis berodde detta på att de stora leverantörerna valde att inte implementera dem i sina produkter.

### 2.3.3 Openflow och nätverksoperativsystem

År 2007 var många principer som ligger till grund för mjukvarudefinierade nätverk redan etablerade i forskarvärlden, men det fanns ännu ingen mogen lösning för att programmera nätverk som fungerade i praktiken. Det fanns även en slitning mellan principer för att skapa öppna, helt programmerbara nätverk och en pragmatism för att skapa praktiskt användbara nätverkslösningar.

Utifrån förutsättningarna vid tiden tog ett forskarteam på Stanford fram det första utkastet till protokollet Openflow. Protokollet utgick till stor del från hur de hårdvarubaserade datavägarna i nätverksutrustningen gick att styra, vilket gjorde det möjligt att implementera Openflow i befintliga produkter. Den första fastslagna versionen av Openflow-standarden släpptes i december år 2009.

Många nätverkskomponenter gör i princip samma sak. De jämför ett fält i paket-huvudet med ett värde och utför en operation utifrån resultatet av jämförelsen. På konceptnivå generaliserar Openflow nätverkskomponenter genom att expandera jämförelserna så att besluten kan baseras på flera fält som är dynamiskt valbara, något som görs genom så kallade flödestabeller. Med olika innehåll i flödes-tabellerna kan en generell hårdvarukomponent därigenom agera som till exempel en switch, en router eller en brandvägg (Feamster m.fl., 2014; Kreutz m.fl., 2015). Notera att alla nätverksfunktioner inte kan styras genom OpenFlow, då flödestabellerna huvudsakligen bara kan uttrycka funktioner som innebär jämförelser med olika fält i paketen.

Parallellt med utvecklingen av Openflow pågick forskning på det som kallas nätverksoperativsystem[11], det vill säga en slags styrfunktion med helhetsansvar

[11] Nätverksoperativsystem (eng. network operating systems) är en lite problematisk term då den används för två helt olika funktionsnivåer i nätverk. Dels används den i denna betydelse (en komponent med helhetsansvar för nätverkets tillstånd), dels används den för operativsystem som är specialiserade för användning i enskilda (vanligtvis fysiska) nätverksenheter. På grund av otydligheten undviker vi därför termen i denna rapport.

för tillståndet i hela nätverket. Nätverksapplikationer kan därmed delegera denna del till nätverksoperativsystemet och i stället vara helt fokuserade på det nätverksbeteende som de var tänkta att styra.

Några av de händelser som utmärker denna tidsperiod är:

- Openflow etablerade sig som en praktiskt användbar standard med relevant genomslag på marknaden.
- Protokoll såsom Openflow gjorde det möjligt att generalisera nätverkskomponenter genom att låta dem basera flödesbesluten på flera av paketens fält.
- Utvecklingen av konceptet nätverksoperativsystem med ett helhetsansvar för funktionen hos nätverket.

Utvecklingen av Openflow fokuserade i mångt och mycket på att skapa en praktiskt användbar lösning som gick att implementera utan alltför omfattande ändringar i befintliga produkter. Samtidigt var marknadens olika aktörer redo att förändra hur nätverksfunktionerna styrdes, vilket öppnade för en bred acceptans av Openflow. Genom att branschen anammade Openflow tog utvecklingen fart även kommersiellt.

## 2.4 Protokoll och protokollhierarkier

Alla nätverk bygger på att det är tydligt specificerat hur olika enheter ska kommunicera. Sådana specifikationer brukar kallas för *protokoll* och utgör konventioner över hur kommunikationen ska gå till. Ett protokoll specificerar vanligtvis flera olika aspekter av kommunikationen och hur den ska hanteras, exempelvis:

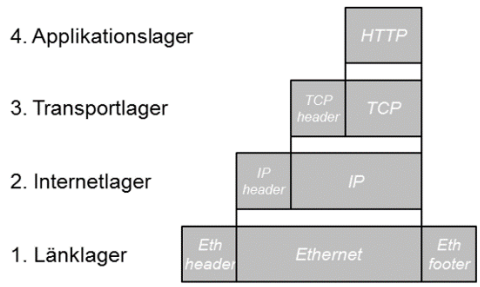
- utformning av meddelanden
- tolkning av innehåll i meddelanden
- relationer mellan meddelanden
- tillståndsmaskiner
- parametrar (såsom tidsgränser)
- felhantering.

Många av de protokoll som vardagsvis används i datorer är väl etablerade och standardiserade genom öppna standardiseringsorgan. Protokollen används dock sällan ensamma i datornätverk, utan samarbetar i så kallade *protokollstackar* (eng. protocol stacks) för att lösa kommunikation mellan två applikationer i olika enheter.

Diskussioner om nätverk, nätverksprotokoll och hur olika protokoll relaterar till varandra kan bli svåra att följa utan en beskrivningsmodell. En sådan modell är *internetprotokollsviten* (eng. internet protocol suite) (Braden, 1989). Den är utformad för att spegla relationer mellan de så kallade internet-protokollen, det

vill säga de protokoll som används på internet såsom *Internet Protocol* (IP), *Transport Control Protocol* (TCP) och *Hypertext Transfer Protocol* (HTTP).

Internetprotokollsviten består av fyra lager så som visas i figur 7, där varje lager exemplifieras med de protokoll som typiskt används för att hämta en webbsida. Internetprotokollsviten beskriver ansvarsområde för respektive lager samt hur de olika lagren interagerar med varandra så att protokollen ska kunna hantera alla relevanta aspekter av kommunikationen i samverkan. Det översta lagret ligger närmast applikationen medan lagret längst ner inkluderar det fysiska medium som kommunikationen sker över.[12] Modellen bygger på att protokollen på respektive lager nyttjar tjänster som tillhandahålls av lagret under och att protokollen erbjuder tjänster till lagret över.




Figur 7. Internetprotokollsvitens fyra lager med exempel på protokoll på respektive lager.

Internetprotokollsviten är utformad för att spegla just internetprotokollens relationer och gör inte anspråk på att vara en generell förklaringsmodell. Det innebär att sviten ofta inte är applicerbar på klassiska protokoll för helt andra ändamål, exempelvis inom industriell automation eller telekom, även om de senaste årens konvergens mellan olika branscher allt mer suddar ut dessa skillnader. OSI-modellen är utformad för att vara en betydligt mer generell förklaringsmodell och delar upp protokollen i sju lager (International Organization for Standardization, 1994). För denna rapport är dock OSI-modellen onödigt komplex, varför internetprotokollsvitens uppdelning är mer lämplig.

Nedan följer korta beskrivningar av respektive lager i internetprotokollsviten.

### 2.4.1 Lager 1 – Länklager

Då internetprotokollsviten inte definierar några egna protokoll på länklagret (eng. link layer) så definieras detta enbart genom ett antal olika krav på länklagrets

[12] Internetprotokollsviten definierar inga egna länklagerprotokoll, utan lämnar detta till andra standarder. Däremot finns det gott om referenser till Ethernet, såväl i RFC 1122 (Braden, 1989) som i RFC 1042 (Postel & Reynolds, 1988). Det är således protokoll utanför internetprotokollsviten som definierar de fysiska medium som används.

protokoll. Kraven sätter ramarna för protokollen på länklagret så att dessa ska kunna nyttjas av de övre lagren i modellen.

I praktiken innebär kraven att länklagret ska erbjuda en grundtjänst som består av överföring av paketorienterad information mellan två eller flera enheter. Överföringen inkluderar paketeringen av data, hur de kommunicerande entiteterna adresseras samt hur överföringen mellan enheterna går till (oavsett om detta sker över ett fysiskt eller virtuellt medium). Länklagret är typiskt tänkt att vara lokalt. *Ethernet*-familjen är vanligt förekommande protokoll på länklagret.

### 2.4.2 Lager 2 – Internetlager

Internetlagret (eng. internet layer) är det protokollager som ligger närmast över länklagret. Lagret ansvarar för adressering på global nivå. Lagret inkluderar mekanismer för att överföra paket mellan enheter som potentiellt ligger på olika (men sammankopplade) nätverk med stora geografiska avstånd.

IP och *Internet Control Message Protocol* (ICMP) utgör två av de vanligaste protokollen på internetlagret. IP hanterar den globala adresseringen på internet och finns i två varianter, *IPv4* och *IPv6*, med olika adressrymder. *Internet Protocol Security* (IPsec) är också ett protokoll på internetlagret som dessutom implementerar säkerhetsmekanismer såsom kryptering och autentisering.

### 2.4.3 Lager 3 – Transportlager

Transportlagret (eng. transport layer) erbjuder tjänster för att skapa datakanaler mellan de kommunicerande parterna. Datakanalerna kan antingen vara *förbindelseorienterade* (eng. connection oriented) eller *förbindelselösa* (eng. connectionless). Ett förbindelseorienterat protokoll brukar ha flera egenskaper som förenklar kommunikationen för ovanliggande lager, exempelvis att kanalen upprätthålls så länge som kommunikation ska pågå, att paket som inte kommer fram sänds om och att mottagen data garanterat levereras i rätt ordning till överliggande lager hos mottagaren.

I förbindelselösa protokoll utgör transportlagret i regel en enkel överföringsmekanism där paket skickas utan de extra funktioner som räknas upp för förbindelseorienterade protokoll.

TCP utgör ett förbindelseorienterat protokoll på transportlagret. Motsvarande förbindelselösa protokoll är *User Datagram Protocol* (UDP).

### 2.4.4 Lager 4 – Applikationslager

Applikationslagret (eng. application layer) ligger närmast användarapplikationen och kapslar in den applikationsspecifika kommunikationen så att den går att sända över de lägre lagrens protokoll. Det finns en uppsjö olika applikationsprotokoll som används för exempelvis överföring av webbsidor, fjärrstyrning av

datorer, synkronisering av tid och rapportering av mätdata. Därtill finns ett antal applikationsprotokoll som är mer inriktade på funktionen hos själva nätverket, såsom nätverksadministration, namnuppslagning och adresstilldelning.

Funktionerna hos applikationslagrets olika protokoll bestäms i stor utsträckning av behoven hos de associerade applikationerna. Exempelvis ingår ofta funktionalitet för att identifiera och autentisera kommunicerande parter där detta är viktigt för applikationerna.

Applikationen, det vill säga programvaran, är inte detsamma som applikationslagret även om applikationen ofta implementerar applikationslagrets protokoll. Applikationen faller utanför modellens lagerstruktur och innehåller dessutom i regel betydligt mer funktionalitet än bara applikationsprotokollet.

Några exempel på applikationsprotokoll är *Dynamic Host Configuration Protocol* (DHCP), *Domain Name System* (DNS) och *File Transfer Protocol* (FTP). Ytterligare exempel är HTTP och dess utökning *Hypertext Transfer Protocol Secure* (HTTPS). De tidigare nämnda protokollen Openflow och Forces är applikationslagerprotokoll.

# 3 Mjukvarudefinierade nätverk

Syftet med mjukvarudefinierade nätverk är att göra nätverk programmerbara. Därför införs en separation mellan styr- och dataplanet och det läggs ett tydligt fokus på öppna gränssnitt. Dessa förändringar gör att styrningen i nätverket kan centraliseras, vilket i sin tur kan förenkla nätverkspolicyhanteringen. Genom att kombinera olika tekniker, såsom mjukvarudefinierade nätverk och virtualisering, underlättas automatisering av IT-infrastrukturens hantering.

Organisationen, Internet Research Task Force (IRTF) beskriver mjukvarudefinierade nätverk som ”*en ansats till programmerbara nätverk som stödjer separation av data- och styrplan via standardiserade gränssnitt*”[13]. Kreutz m.fl. (2015) definierar att mjukvarudefinierade nätverksarkitekturer bör ha följande egenskaper.

- Styr- och dataplanet separeras och implementeras i separata komponenter. Styrplanet flyttas ur nätverksfunktioner, vilka endast vidarebefordrar nätverkstrafik.
- Nätverkets styrningslogik centraliseras till styrenheter.
- Applikationer kan programmera nätverket via styrenheter.
- Vidareförmedlingsregler är flödesbaserade.

Ovanstående definitioner innebär att mjukvarudefinierade nätverk har en specifik och avgränsad funktion – nämligen att centralt administrera och styra hur nätverksfunktionerna tar beslut om vidarebefordring av trafik. Detta innebär att definitionerna av mjukvarudefinierade nätverk exkluderar många viktiga nätverksadministrativa uppgifter, såsom konfiguration, driftsättning och övervakning av nätverksfunktioner. Mjukvarudefinierade nätverk utgör därmed ett komplement till befintliga nätverksadministrativa lösningar, där mjukvarudefinierade nätverk har som uppgift att styra trafikflöden i nätverket (Haleplidis m.fl., 2015).

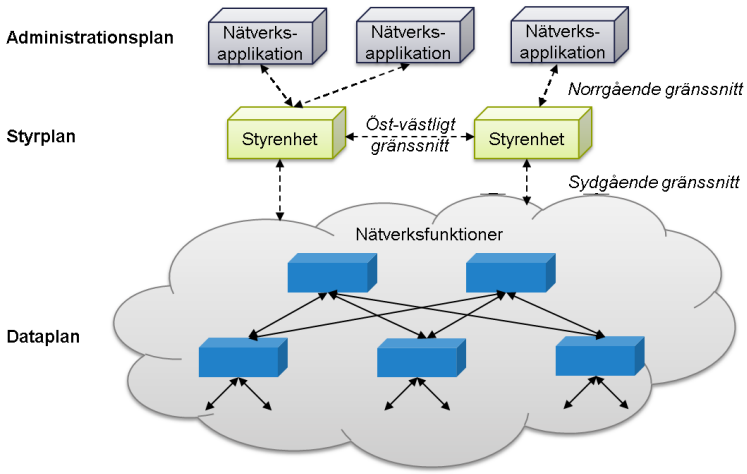
I detta kapitel beskrivs vad mjukvarudefinierade nätverk är, vad de kan användas till samt i vilka system de kan användas.

## 3.1 Funktionsplan och gränssnitt

I mjukvarudefinierade nätverk implementeras nätverkens tre funktionsplan i separata komponenter. Administrationsplanet utgörs av nätverksapplikationer. Styrplanet implementeras i en, eller flera logiskt centraliserade, styrenheter. Dataplanet implementeras i nätverksfunktioner. Mellan enheterna finns väl-

[13] ”A programmable networks approach that supports the separation of control and forwarding planes via standardized interfaces.” (Haleplidis m.fl., 2015).

definierade protokoll för såväl norrgående som sydgående och öst-västligt gränssnitt, vilket illustreras i figur 8.




Figur 8. Funktionsplan, gränssnitt och komponenter i mjukvarudefinierade nätverk

I mjukvarudefinierade nätverk ansvarar styrplanet för att implementera nätverks-policyer. Parametrar såsom tidpunkt, länkbelastning, topologi eller typ av paket kan påverka vilken policy som bör gälla. Styrenheter kan över tid distribuera nya flödestabeller till berörda nätverkskomponenter i enlighet med en policy (Kousalya m.fl., 2017, s. 39; Kreutz m.fl., 2015; Raj & Raman, 2019, s. 25).

### 3.1.1 Administrationsplan

Administrationsplanet består av nätverksapplikationer som definierar nätverks-policyer, vilka slutligen ligger till grund för hur paket bearbetas och styrs i nätverket. Nätverksapplikationer bestämmer vilka nätverkspolicyer som ska gälla i nätverket, men det är upp till styrenheten att realisera en nätverkspolicy. Att nätverksapplikationer tillåts bestämma nätverksbeteende utan att behöva implementera beteendet är en viktig abstraktion i mjukvarudefinierade nätverk.

Nätverksapplikationer kommunicerar med styrplanet genom det norrgående gränssnittet. Över detta gränssnitt kan applikationerna skicka nätverkspolicyer och hämta information rörande bland annat nätverkets topologi (Kreutz m.fl., 2015).

### 3.1.2 Styrplan

I ett mjukvarudefinierat nätverk implementeras styrplanet av styrenheter. Även om styrplanet kan bestå av en enda styrenhet används ofta flera samverkande styrenheter för att exempelvis få bättre redundans och skalbarhet. I en distribuerad uppsättning kommunicerar styrenheterna med varandra via öst-

västliga gränssnitt som går inom styrplanet. Vidare måste styrenheterna inte implementeras på separata hårdvarukomponenter, utan kan driftsättas tillsammans med exempelvis switchar och applikationer på en och samma server (Kreutz m.fl., 2015).

I mjukvarudefinierade nätverk implementeras styrplanet och dataplanet i separata komponenter. Det innebär att nätverksfunktionernas styrplan flyttas till styrenheterna, vilket förvandlar förmedlingsfunktioner i dataplanet till simpla komponenter som endast utför styrenhetens instruktioner. Styrenheten översätter nätverkspolicyer till flödesregler och distribuerar dessa till dataplanet. Separationen av styr- och dataplan leder således till att nätverksapplikationer i administrationsplanet inte behöver interagera direkt med förmedlingsfunktioner i dataplanet.[14]

I mjukvarudefinierade nätverk samlar styrplanet in information från nätverks-funktioner, aggregerar informationen och tillhandahåller slutligen informationen till nätverksapplikationer via det norrgående gränssnittet. Den aggregerade informationen kan exempelvis innehålla status för länkar och switchar samt en nätverksgraf över nätverkets topologi. Nätverksgrafen med tillhörande information, som inkluderar nätverkets samtliga nätverksfunktioner, deras relationer och trafikflöden, kallas vanligtvis för en *global vy* över nätverket (Jain m.fl., 2019; Kreutz m.fl., 2015; Sandhya m.fl., 2017).

Styrplanet påverkar hur paket styrs genom att distribuera instruktioner via det sydgående gränssnittet. Protokollet vid det sydgående gränssnittet kan antingen bygga på en *imperativ* eller *deklarativ* modell. Den imperativa modellen betyder i princip att styrenheten i detalj beskriver hur respektive paket ska hanteras av dataplanet, vilket i praktiken innebär att dataplanet inte behöver ta några egna beslut om paketen. Den imperativa modellen ger centraliserad detaljstyrning eftersom styrningen helt dikteras av styrenheten. I den deklarativa modellen överförs policyer på applikations- eller flödesnivå till dataplanet, vilket lämnar en viss grad av beslutsfattande till nätverkskomponenten. Den deklarativa modellen medför därmed att intelligensen i nätverket distribueras i viss utsträckning (Latif m.fl., 2020).

[14] Att nätverksapplikationer inte direkt styr eller samlar information från nätverksfunktioner är en abstraktion som Kreutz m.fl. (2015) kallar för ”the distribution abstraction”.

Det finns ingen vedertagen standard för kommunikationen via det norrgående gränssnittet. Kommunikationen kan exempelvis ske via produktspecifika API:er där policyerna överförs. Policyerna kan då beskrivas i speciella programmeringsspråk som tagits fram specifikt för att uttrycka nätverkspolicyer (Kreutz m.fl., 2015).

### 3.1.3 Dataplan

Dataplanet består av nätverksfunktioner som inspekterar och vidareförmedlar paket i nätverket (Kreutz m.fl., 2015). Beroende på vilken protokollmodell som används vid det sydgående gränssnittet är nätverksfunktioner mer eller mindre intelligenta. I en strikt imperativ modell har nätverksfunktioner ingen intelligens alls och räknar själva inte ut var paket ska vidareförmedlas.

Samtliga förmedlingsfunktioner i nätverket bör exponera likadana gränssnitt och styras med samma kommandon eller protokoll. Det sydgående gränssnittet introducerar således en viktig abstraktion i mjukvarudefinierade nätverk, vilken är att styrenheten inte känner till vilken hårdvara förmedlingsfunktionerna består av. Genom det sydgående gränssnittet tar varje förmedlingsfunktion emot vidareförmedlingsregler från styrenheten. Förmedlingsfunktioner förmedlar även information till styrenheten genom det sydgående gränssnittet (Kreutz m.fl., 2015).

## 3.2 Nätverksvirtualisering

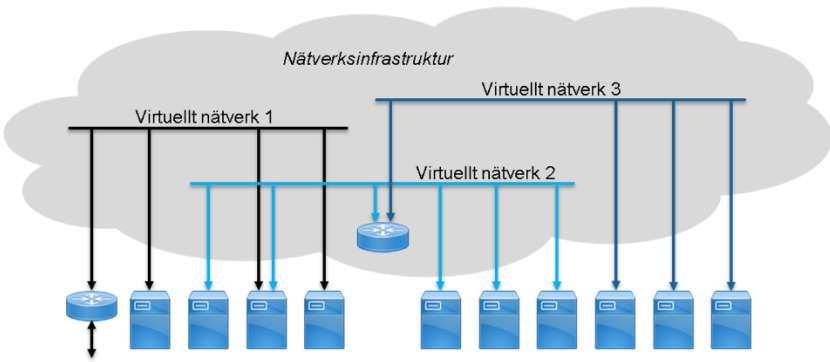
Virtualisering inför en abstraktion mellan driftsatta mjukvarubaserade system och hårdvaran som systemet körs på. Detta medför flera fördelar, bland annat att hårdvarans nyttjandegrad kan öka. Server- och nätverksvirtualisering har gjort att olika typer av nätverkskomponenter kan driftsättas på generell hårdvara[15], i stället för att de kräver skräddarsydd hårdvara (Raj & Raman, 2019, s. 31). Mjukvarudefinierade nätverk beskrivs ibland som en möjliggörare för bland annat virtuella nätverksfunktioner och nätverksvirtualisering (Jain m.fl., 2019, s. 9; Kousalya m.fl., 2017, s. 41).

Nätverksvirtualisering handlar om att exempelvis låta olika kunder, system eller applikationer få egna virtuella nätverk som är till synes oberoende av varandra och den underliggande nätverksinfrastrukturen. Flera virtuella nätverk kan därmed dela på ett och samma fysiska nätverk, med ett minimum av interaktion dem emellan. De virtuella nätverken kallas ibland för överläggsnätverk (eng. overlay network) som transporteras över ett underlagsnätverk (eng. underlay network) som i sin tur består av exempelvis det fysiska nätverket. Dessa termer

[15] Nätverkskomponenter såsom routrar, switchar, brandväggar och intrångsdetektionssystem kan alla driftsättas på exempelvis servar med x86-arkitektur.

brukas dock inte exklusivt för nätverksvirtualisering, utan nyttjas även när andra tekniker används för att kapsla in ett nätverk över ett annat nätverk, såsom i fallet med virtuella privata nätverk (VPN)[16].

Figur 9 illustrerar hur virtuella nätverk kan dela på en underliggande nätverksinfrastruktur och hur de virtuella nätverken kan hållas isär eller kopplas samman på liknande sätt som det görs i traditionella, fysiska nätverk. Användningen av nätverksvirtualisering har inget direkt beroende eller relation till om den underliggande nätverksinfrastrukturen är byggd med fysiska eller virtuella komponenter. I stället är nätverksvirtualisering närmast ett användningsfall för nätverken, där infrastrukturens olika komponenter – exempelvis mjukvarudefinierade nätverk – används för att skapa de virtuella nätverken.




Figur 9. Virtuella nätverk ovanpå en fysisk nätverksinfrastruktur.

Zhang m.fl. (2017) beskriver att nyttjandegraden för hårdvaruresurser kan förbättras genom att skapa flera virtuella nätverk som delar på samma hårdvaruresurs. Vidare kan skräddarsydda virtuella nätverk skapas för att möta tjänster med olika behov bättre. Slutligen beskrivs att det är kostnadseffektivt att rulla ut nya tjänster med hjälp av nätverksuppdelning, eftersom tjänsterna först kan testas i ett eget isolerat nätverk.

## 3.3 Lastbalansering

Lastbalansering handlar om att fördela nätverkstrafik jämnt över länkar för att undvika under- eller överutnyttjande av hårdvaran. Jämnt fördelade flöden leder i regel till bättre prestanda i nätverken då det exempelvis kan göra så att hårdvaran nyttjas i högre grad. Dessutom kan lastbalansering optimera omfördelning av

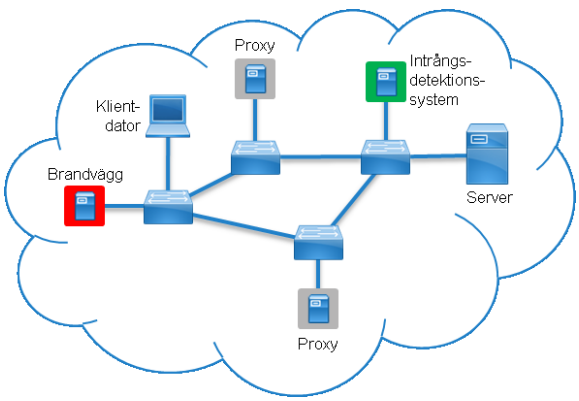
[16] Namnet till trots, är inte VPN ett specialfall av nätverksvirtualisering. VPN är en teknik för att skapa krypterade tunnlar över (osäkra) nätverk, till exempel mellan en hemarbetande användare och ett företags interna nätverk.

trafiken när nätverket förändras, till exempel när nätverkets topologi förändras för att en länk eller nätverksfunktion slutar fungera (Akyildiz m.fl., 2014).

I ett mjukvarudefinierat nätverk kan styrenhetens kännedom om nätverket användas som beslutsunderlag för hur paket ska flöda i nätverket. Exempelvis är nätverkets aktuella topologi en viktig parameter för att kunna leda om trafik på ett smart sätt. Det är upp till applikationer att definiera policyer för lastbalansering och det är styrenhetens uppgift att skapa vidareförmedlingsregler som ger bra lastbalansering samt distribuera dessa regler i dataplanet. Med hjälp av mjukvarudefinierade nätverk kan styrenheten leda om trafik utan att en nätverksadministratör behöver hantera varje nätverkskomponent för sig (Akyildiz m.fl., 2014).

## 3.4 Tjänstekedjor

När flera olika funktioner såsom brandväggar, intrångsdetektorer, adressöversättningar och proxyer används tillsammans i en specifik sekvens skapas en så kallad *tjänstekedja* (eng. service chain). Tidigare byggdes tjänstekedjor genom att de ingående komponenterna kopplades ihop i kedjan för att sedan introduceras på lämplig plats i nätverket. Dessa fysiskt skapade tjänstekedjor saknade typiskt flexibilitet och krävde ofta betydande konfigurationsarbete för att fungera väl (John, 2013). Numera byggs tjänstekedjor ofta genom konfiguration av flödesvägar som kan gå genom såväl fysiska som virtuella nätverksfunktioner.




Figur 10: Ett nätverk med en brandvägg (röd), två proxyer (grå) och ett intrångsdetektionssystem (grön).

Om det finns flera likadana nätverksfunktioner i ett nätverk kan det finnas flera vägar som flöden kan ledas för att gå genom en viss tjänstekedja. Exempelvis kan trafik följa olika vägar när den skickas mellan de två datorerna i figur 10. Således kan optimeringar i flödets färdväg göras baserat på vilka krav och övriga trafikflöden som finns i nätverket. Om det exempelvis uppstår överbelastning på

vissa länkar i nätverket kan det vara bättre att ta en längre väg genom en tjänstekedja för att bättre fördela trafiken över länkar och nätverksfunktioner. Med mjukvarudefinierade nätverk kan en nätverksadministratör skapa policyer för att försöka optimera trafikflödet samtidigt som trafiken leds genom en tjänstekedja i rätt ordning (Guo m.fl., 2016).

Om styrenheten är medveten om vilka tjänster som tillhör vilka flöden, kan flöden som tillhör en tjänst styras genom en tjänstekedja medan ett annat flöde leds genom en annan tjänstekedja (Li, 2018). Om användarens behov förändras kan en ny policy driftsättas som styr om flödena genom nya tjänstekedjor (Li & Chen, 2015).

## 3.5 Användningsområden

I detta avsnitt presenteras några områden där mjukvarudefinierade nätverk har använts eller skulle kunna användas. Författarnas uppfattning är att användningen av mjukvarudefinierade nätverk har kommit långt inom ibland annat datacenter och *Wide Area Networks* (WAN). Exempelvis har Google en implementation där mjukvarudefinierade nätverk används för att förbinda datacenter över ett WAN[17]. Mjukvarudefinierade nätverk kan komma att spela en viktig roll i framtida mobilnätverk, men den utvecklingen verkar just nu befinna sig i ett forskningsstadium (Kamath m.fl., 2020; Reith, 2019).

Mjukvarudefinierade nätverk är ännu relativt ovanligt i lokala nätverk (eng. Local Area Network, LAN) då tekniken huvudsakligen riktar sig mot datacenterlösningar och infrastrukturoperatörers nätverk. Det finns dock åtskilliga källor som menar att tekniken kan komma att användas mer även i lokala nätverk i framtiden. Det finns en pågående trend mot så kallade mjukvarudefinierade LAN (eng. Software-Defined LAN, SD-LAN), där mjukvarudefinierade nätverk och virtuella nätverksfunktioner används i ökande grad även i LAN-sammanhang (se exempelvis Cooney, 2019; Extreme Networks, 2019; Orange, 2019; T-Systems, 2018).

### 3.5.1 Datacenter

Ett datacenter behandlar stora mängder data och stora mängder nätverkstrafik. Operatörer av datacenter tillhandahåller tjänster[18] till kunder som ofta har olika typer av behov. Nya trender kan förändra behoven över tid. Exempelvis kan tjänstekvaliteten, bandbredden eller prestandan som en kund förväntar sig kan variera över tid (Raj & Raman, 2019, s. 18 och 68).

[17] Jain m.fl. (2019) nämner även att företagen Google, AT&T, Microsoft, Intel, Verizon, Ericsson, Huawei och Equinox använder mjukvarudefinierade nätverk på olika sätt, främst i datacenter.

[18] Exempelvis lagring, beräkningskraft eller nätverk i molnet.

Kousalya m.fl. (2017, s. 42) lyfter fram att mjukvarudefinierade nätverk i datacenter kan öka nyttjandegraden av hårdvaran genom att distribuera flöden jämnare över länkar. Vidare beskriver Raj & Raman (2019, s. 84) att mjukvarudefinierade nätverk kan driftsätta policyer dynamiskt beroende på situation för att möta olika kunders behov. Prestandabehov hos kunder kan skifta över tid, då exempelvis en applikation plötsligt kan kräva mer bandbredd. När detta inträffar kan styrenheten konfigurera om flödesregler i nätverkets förmedlingsfunktioner baserat på en policy så att kundens nya behov tillfredsställs. Slutligen menar Raj & Raman (2019, s. 84) att flödesregler i samtliga förmedlingsfunktioner kan konfigureras via styrenheten, vilket är smidigare än att konfigurera förmedlingsfunktioner var för sig.

### 3.5.2 SD-WAN

Syftet med ett WAN är att koppla ihop LAN över geografiskt spridda platser, vilket kan vara viktigt såväl för organisationer som för datacenter. *Multiprotocol Label Switching* (MPLS) är den teknik som traditionellt använts för att bygga upp WAN. I ett MPLS-nätverk märks varje inkommande paket i nätverket med en etikett som bestäms utifrån information i paketet och andra parametrar. För att använda MPLS skapar administratörerna policyer som beskriver vilken förutbestämd väg paketen ska färdas i nätverket. Varje router i nätverket vidarebefordrar därefter paketen baserat på etiketten som paketet tilldelats och de policyer som är uppsatta.[19] Således behöver routrarna inte räkna ut hur paketet ska vidarebefordras, vilket sparar tid och beräkningsprestanda (Li, 2018).

Genom dess egenskaper kan MPLS användas för att säkerställa tillförlitlig och snabb transport av trafik över WAN (Kousalya m.fl., 2017; Wood, 2017). Nackdelen är att MPLS är tidskrävande att konfigurera[20], vilket gör det dyrt att använda (Kousalya m.fl., 2017, s. 48; Li, 2018, s. 28; Wood, 2017). Idag finns det visst verktygsstöd för att underlätta konfigurering av MPLS[21], men mycket av arbetet är fortfarande i stor utsträckning manuellt.

Mjukvarudefinierade WAN (eng. Software Defined WAN, SD-WAN) innebär en utökning som går utöver WAN med mjukvarudefinierade nätverk, då det snarare är en virtualiserad infrastruktur spridd över WAN. I en sådan infrastruktur hos en infrastrukturleverantör kan exempelvis virtuella funktioner placeras ända ut i den utrustning som placeras hos kunderna[22] (Reith, 2019). Idag är det till exempel vanligt att kundernas data lagras i flera datacenter som är geografiskt spridda

[19] Arkitekturen i MPLS beskrivs i RFC 3031, se https://tools.ietf.org/html/rfc3031 [läst 2020-10-27].

[20] Jain m.fl. (2019, s. 15) beskriver att MPLS konfigureras på en ”box-by-box” basis.

[21] Till exempel genom scriptning i verktyget Ansible från Red Hat (https://www.ansible.com/).

[22] Ändutrustningen hos kunderna brukar kallas CPE efter engelskans *Customer Premises Equipment*, det vill säga utrustning på kundens område.

samtidigt som många applikationer ligger i molnet, vilket ökar intresset för SD-WAN (Reith, 2019).

Wood (2017a och 2017b) beskriver att SD-WAN kan koppla ihop nätverk från geografiskt skilda platser. Ett SD-WAN skapar överläggsnätverk ovanpå andra transportnätverk, exempelvis internet och MPLS[23]. Oavsett vilket typ av transportnätverk som används har varje plats en nod vid utkanten av sitt nätverk. Varje nod ser till att flöden skickas till andra noder på ett tillförlitligt sätt över ett lämpligt transportnätverk. Exempelvis kan en nod välja en länk med bättre eller sämre reliabilitet[24] baserat på hur nätverket ser ut just nu och vilken prioritet en applikation har. Styrenheter ser till att policyer implementeras och kommunicerar med noderna i utkanten av varje nätverk.

Li (2018, s. 34–35) menar att den främsta drivkraften för SD-WAN är intelligent trafikstyrning. Styrenheten känner till nätverkstopologi och trafikinformation vilka kan användas för att styra nätverksflöden på ett smart sätt, för att exempelvis undvika att överbelasta länkar. Omfördelning av trafiken utifrån tillgänglig bandbredd kan öka nätverkets nyttjandegrad och förbättra användarnas tjänstekvalitet. Wood (2017a) menar att det är enkelt att koppla nätverk på nya platser till ett befintligt datacenter. Istället för att förlita sig på en vanlig internetanslutning kan ett SD-WAN då optimera flödens rutter och tillse att anslutningar fungerar. Kousalya m.fl. (2017) lyfter fram att SD-WAN garanterar anslutning och bra prestanda till molnapplikationer oavsett på vilken geografisk plats som applikationerna finns. SD-WAN är oberoende av vilket underlagsnätverk som används, exempelvis kan SD-WAN nyttjas över internet eller mobilnätverk (Reith, 2019, s. 126; Wood, 2017).

### 3.5.3 Mobilnätverk

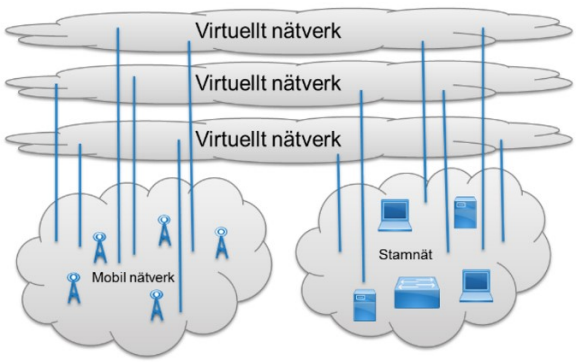
Mobilnätverk hanterar en allt större och mer heterogen mängd data än tidigare och denna utveckling ser ut att fortsätta (Barakabitze m.fl., 2020). Detta beror på att användare i högre utsträckning använder olika typer av tjänster som ställer höga krav på nätverket. Det innebär en ökad belastning på mobilnätverk men även på stamnätverk dit mycket trafik skickas. Den nya generationens mobilnätverk, 5G, har mycket varierande krav, bland annat på kvalitet, fördröjning, säkerhet och skalbarhet. Reith (2019, s. 115) menar det finns konsensus om att mjukvarudefinierade nätverk och virtuella nätverksfunktioner kommer vara nyckelspelare i automatiseringen av 5G.

[23] MPLS skapar också överläggsnätverk fast på en underliggande, typiskt fysisk, nätverksinfrastruktur. SD-WAN över MPLS är således ett nätverk med dubbla överläggsnivåer.

[24] Reliabilitet i form av exempelvis paketförlust och jitter. Beroende på vilka transportnätverk som finns tillgängliga kan noden välja det som den finner bäst lämpligt.

*Network slicing*[25] är en speciell form av nätverksvirtualisering som riktar in sig på stomnätverken som bygger upp mobilnätverken (Kamath m.fl., 2020). Network slicing är specifikt framtaget för att mobiloperatörerna ska kunna hantera virtuella nätverk med olika garantier på tjänstekvalitet för affärskunder, baserat på respektive kunds servicenivåavtal (GSM Association, 2017).

Med hjälp av nätverksvirtualisering kan det fysiska nätverket delas upp i separata virtuella nätverk och skräddarsys för en typ av tjänst, se figur 11. Network slicing beskrivs som en viktig del i det nya 5G-nätverket för att kunna hantera olika typer av tjänster med varierande krav. Styrenheten i mjukvarudefinierade nätverk kan bidra med att skapa och hantera virtuella nätverk samt driftsätta policyer i vardera virtuellt nätverk (Barakabitze m.fl., 2020).




Figur 11: Nätverksuppdelning i mobil- och stamnät

Zhang m.fl. (2017) föreslår till exempel ett ramverk där Network Slicing och mjukvarudefinierade nätverk kombineras för att flexibelt och dynamiskt dela upp infrastrukturen i virtuella nätverk. Varje virtuellt nätverk kan spänna över både stam- och accessnätverket[26]. I en sådan lösning kan mjukvarudefinierade nätverk bidra med att utföra nätverksuppdelningen på ett flexibelt sätt, där virtuella nätverksfunktioner kan driftsättas och tjänstekedjor skapas efter behov. Vidare beskriver Zhang m.fl. (2017) att varje virtuellt nätverk kan ha en egen styrenhet och därmed egna skräddarsydda policyer.

Mjukvarudefinierade nätverk har potential att minska den framtida ökningen av mängden data i mobilnäten. Exempelvis presenterar Lv och Xiu (2020) en lösning där mjukvarudefinierade nätverk och virtualiserade nätverksfunktioner

[25] Network Slicing är ett relativt nytt begrepp som ännu inte har en vedertagen översättning på svenska. Närmast ligger ”nätverksuppdelning”, men då den termen är lätt att feltolkas har vi valt att använda den engelska termen.

[26] Ett accessnätverk är det nätverk av kablage och utrustning som finns mellan en abonnent och en telestation.

används tillsammans med så kallad *Multi-Access Edge Computing*[27] (MEC) för att minimera mängden data som behöver skickas över stomnätverket. Kamath m.fl. (2020) presenterar ett annat exempel där en kombination av Network Slicing och mjukvarudefinierade nätverk används för att sammanställa trafikens karakteristik i varje enskild skiva i nätverket. Genom simuleringar visar Kamath m.fl. (2020) hur nätverksstatistik från respektive skiva kan användas för att beräkna en trafikmodell för hela nätverket. Trafikmodellen kan sedan användas för att anpassa policyerna för att öka hårdvarans utnyttjandegrad och förbättra kundernas tjänstekvalitet.

[27] Multi-Access Edge Computing (MEC) (tidigare Mobile Edge Computing) innebär att datorkraft och tjänster placeras ut nära accessnätet. Exempelvis kan molntjänster och andra applikationer ligga mycket nära radionätet i mobiltelefonisystemen. För mer information, se https://www.etsi.org/technologies/multi-access-edge-computing.

# 4 Nätverksfunktioner

*Nätverksfunktioner* (eng. network functions, NF) är ett samlingsbegrepp som inkluderar alla typer av avgränsade, funktionella byggstenar i ett datornätverk. Nätverksfunktioner har typiskt väldefinierade yttre gränsytor och väldefinierade beteenden.[28] Ett nätverk byggs upp genom att strukturerat koppla samman olika typer av nätverksfunktioner för att nå den önskade funktionen för nätverket som helhet.

Typiska exempel på nätverksfunktioner är switchar och routrar, det vill säga *förmedlingsfunktioner* som styr paketflöden över de olika länkarna i nätverket utifrån en uppsättning flödesregler. Därtill finns säkerhetsorienterade nätverksfunktioner såsom brandväggar och intrångsdetektorer. Nätverksfunktioner som har säkerhetsfokus utgör *nätverkssäkerhetsfunktioner*.

Nätverksfunktionerna kan antingen implementeras i fysisk miljö eller i virtuell miljö. I de fall då denna distinktion är viktig benämns dessa som *fysiska nätverksfunktioner* (eng. Physical Network Functions, PNF) respektive *virtuella nätverksfunktioner* (eng. Virtual Network Functions, VNF) (ETSI, 2020). Denna indelning är oberoende av vilken typ av nätverksfunktion det avser, det vill säga att det går att virtualisera såväl förmedlingsfunktioner som nätverkssäkerhets-funktioner.

Följande avsnitt går igenom de ovanstående begreppen i mer detalj. Startpunkten för detta är virtualisering, då denna teknik kan appliceras på i stort sett alla tänkbara typer av nätverksfunktioner.

## 4.1 Virtualisering av nätverksfunktioner

Virtualisering av nätverksfunktioner (eng. network function virtualization, NFV) är ett samlingsbegrepp för olika tekniker att implementera nätverksfunktioner i en virtuell miljö. Det kan exempelvis vara en mjukvaruimplementerad switch som körs under en hypervisor[29] i en virtuell servermiljö eller en brandvägg som körs som en virtuell maskin.

Medan nätverksvirtualisering, som togs upp i avsnitt 3.2, utgör en princip för att realisera mjuka nätverksstrukturer på ett underliggande nätverk, så är det de enskilda nätverksfunktionerna som ligger i fokus vid virtualisering av

[28] I denna rapport utgår vi från ETSI:s definition av nätverksfunktion så som den anges i deras arbete kring virtuella nätverksfunktioner. En nätverksfunktion definieras som ett ”*functional block within a network infrastructure that has well-defined external interfaces and well-defined functional behaviour*” (ETSI, 2020).

[29] Hypervisorn är den systemfunktion som skapar en virtuell miljö (den ”mjuka hårdvarumiljön”) som virtuella maskiner och virtuella funktioner körs i.

nätverksfunktioner. Nätverksvirtualisering går att genomföra utan virtuella nätverksfunktioner samtidigt som virtuella nätverksfunktioner inte i sig ger nätverksvirtualisering. Det finns således inget direkt beroende mellan dessa två teknikområden, även om de ofta samexisterar i IT-miljön.

Gränsen mellan vad som är en virtuell nätverksfunktion och vad som utgör en generell mjukvarufunktion är flytande, oavsett om de implementeras på en fysisk nätverksenhet eller i en dator. Det är exempelvis tveksamt om brandväggen i Microsoft Windows är att betrakta som en virtuell nätverksfunktion då den är tätt integrerad med operativsystemet snarare än att köras i en virtualiseringsmiljö.[30] En virtuell switch som implementerats nära hypervisorn i en servermiljö bör nog däremot betraktas som en virtuell nätverksfunktion. Därtill går det att tänka sig fysiska nätverksenheter som utöver sin inbyggda funktion även ger möjlighet att lägga in godtyckliga virtuella nätverksfunktioner såsom brandväggar. I sådana fall uppstår ett slags blandmiljö med både fysiska och virtuella nätverksfunktioner i samma fysiska enhet.

I teorin kan i princip alla olika nätverksfunktioner virtualiseras, men i praktiken kan det finnas begränsningar som omöjliggör virtualisering av specifika nätverksfunktioner. Detta kan exempelvis bero på närhetskrav (att funktionen behöver vara tätt kopplad med en annan funktion som inte går att virtualisera), på prestandakrav som inte kan mötas utan specifik hårdvara eller på säkerhetskrav som inte kan mötas i en virtuell miljö.

En mer utförlig diskussion om virtualisering av IT-system i allmänhet och de risker som virtualisering kan medföra återfinns i FOI-rapporten *Risker med virtualisering av IT-system* (Eidenskog & Karresand 2017).

## 4.2 Förmedlingsfunktioner

Förmedlingsfunktioner[31] (eng. forwarding functions) utgör den kategori av nätverksfunktioner som primärt är till för att dirigera vidare nätverkstrafik.[32] Typiska förmedlingsfunktioner är switchar och routrar, vilka i princip utför

[30] Nätverksfunktioner delas ibland upp i *host based* och *network based*, det vill säga de som implementerats i en dator respektive i en nätverksenhet. Windows-brandväggen är ett exempel på en så kallad *host-based firewall*, men som inte är vare sig en fysisk brandvägg eller en virtuell brandvägg (då den inte befinner sig i en virtualiseringsmiljö).

[31] Den tycks inte finnas en vedertagen term på svenska för engelskans *forwarding function*. I denna text har vi valt *förmedlingsfunktion* då det ligger nära den engelska termen och samtidigt är någorlunda språkligt smidigt på svenska.

[32] Vissa skribenter använder engelskans *forwarding function* strikt för att beskriva funktioner på länklagret (motsvarande switchar) medan andra använder den oavsett protokollager. Vi använder termen på det senare sättet och inkluderar därmed alla typer av förmedlingsfunktioner oavsett protokollager, vilket exempelvis innebär att både switchar och routrar ses som förmedlingsfunktioner i denna rapport.

liknande uppgifter men på olika protokollager.[33] Dessa tar emot paket från inkommande nätverksanslutningar och undersöker en delmängd av de olika fälten för att ta beslut om på vilken eller vilka utgående anslutningar som respektive paket ska skickas vidare.

Traditionellt sett har vidaresändningsbesluten enbart tagits utifrån destinationsadressen på ett specifikt protokollager (Feamster m.fl., 2014). De förmedlingsfunktioner som tar besluten på länklagret, exempelvis utifrån destinationsadressen i Ethernet-paketet, är de som vanligtvis kallas för switchar. Förmedlingsfunktioner som tar besluten på internetlagret, exempelvis utifrån IP-protokollets destinationsadressfält, är de som vanligtvis kallas för routrar. Under årens lopp har det skapats ett antal olika hybridtyper av förmedlingsfunktioner, exempelvis så kallade *lager 3-switchar*[34] som kombinerar switchens funktioner med en delmängd av routerns funktioner.

Mjukvarudefinierade nätverk möjliggör en generaliserad typ av förmedlingsfunktion. Då den centrala styrenheten i det mjukvarudefinierade nätverket har bättre möjlighet att dynamiskt hålla reda på olika trafikflöden i nätverket, kan detta även användas för att detaljstyra paketflödena genom förmedlingsfunktionerna i nätverket. I en sådan lösning kan besluten tas utifrån exempelvis vilka protokoll som paketet innehåller kombinerat med käll- och destinationsadresser för både Ethernet- och IP-lagren samt destinationens TCP-port. Denna typ av förmedlingsfunktion ger därmed en betydande möjlighet till detaljstyrning av vidaresändningsbeslut när de används i mjukvarudefinierade nätverk vilket ger möjlighet att en sådan funktion kan agera som exempelvis router, switch, lastbalanserare eller en enklare brandvägg (Kreutz m.fl., 2015, s. 15).

## 4.3 Nätverkssäkerhetsfunktioner

Nätverkssäkerhetsfunktioner[35] (eng. network security functions, NSF) är nätverksfunktioner som skyddar nätverket i sig eller de data som transporteras i nätverket. Arbetsgruppen *Interface to Network Security Functions* (I2NSF)[36]

[33] Se exempelvis Musa (2018, s. 14–16) och https://www.cloudflare.com/learning/network-layer/what-is-a-network-switch/ [läst 2020-09-29]

[34] Namnet lager 3-switch kommer från OSI-modellens lagerindelning för protokoll. OSI-modellen har sju lager, där lager 1–2 motsvarar internetmodellens länklager och lager 3 motsvarar internetlagret. De fyra återstående lagren motsvarar intenetmodellens transportlager och applikationslager, men där finns inte en lika tydlig koppling mellan indelningen i de två modellerna.

[35] ETSI (2020) tar inte upp någon generell term för nätverkssäkerhetsfunktion. Däremot definierar de två mer specifika termer för *fysisk säkerhetsfunktion* (eng. ”physical security function”) och *virtuell säkerhetsfunktion* (eng. ”virtual security function”). Då denna uppdelning inte alltid är lämplig har vi även med samlingsbegreppet.

[36] Arbetsgruppen I2NSF är ett initiativ för att standardisera ett ramverk som definierar gränssnitt och datamodeller för kommunikation mellan nätverkssäkerhetsfunktionernas styrplan och dataplan.

inom Internet Engineering Task Force (IETF) definierar en nätverkssäkerhetsfunktion som ”en funktion som används för att (1) säkerställa riktighet, konfidentialitet eller tillgänglighet hos nätverkskommunikation, (2) detektera oönskad nätverksaktivitet eller (3) blockera eller mildra effekterna av oönskad aktivitet.”[37].

Nätverkssäkerhetsfunktionerna kan således vara av olika karaktär och det finns ett antal olika typer av säkerhetsfunktioner för användning i nätverk. Några exempel på säkerhetsfunktioner som passar in i definitionen är brandväggar, VPN, intrångsdetektorer (IDS), intrångsskydd (IPS) och datadioder. Funktionen och komplexiteten hos dessa funktioner kan variera stort, exempelvis från en enklare brandvägg som endast undersöker trafikflöden på paket-för-paket-basis, till ett avancerat intrångsdetektionssystem med en distribuerad arkitektur där samverkande övervakningsenheter utspridda i nätverket gemensamt tar beslut baserat på många trafikflöden över lång tid.

Nätverkssäkerhetsfunktioner är traditionellt sett implementerade i fysiska enheter som ansluts på lämpliga punkter i nätverken. Placeringen har ofta varit viktig för att enheterna ska kunna utföra sin funktion, där exempelvis en brandvägg typiskt behövde anslutas i gränslandet mellan det externa och det interna nätverket för att den skulle kunna skydda mot oönskad trafik utifrån.

Nätverkssäkerhetsfunktionerna har huvudsakligen samma allmänna egenskaper och förutsättningar som övriga nätverksfunktioner, vilket bland annat innebär att de i allmänhet går att virtualisera. Säkerhetsaspekterna kring virtualiserings-lagrets implementation blir dock extra viktiga i säkerhetsfunktionerna, vilket innebär att vissa typer av funktioner som typiskt bara återfinns i högsäkerhets-system kan bli poänglösa om de virtualiseras.

Nätverkssäkerhetsfunktioner ger generellt sett liknande administrativa utmaningar som förmedlingsfunktionerna när de används i nätverken. De kräver såväl konfiguration som övervakning och underhåll för att kunna utföra sina uppgifter. Förutom de olika metoder och verktyg för att hantera konfigurationer som beskrivs i kapitel 2 finns det även säkerhetsfokuserade verktyg från företag såsom Clavister[38] och Check Point[39].

---

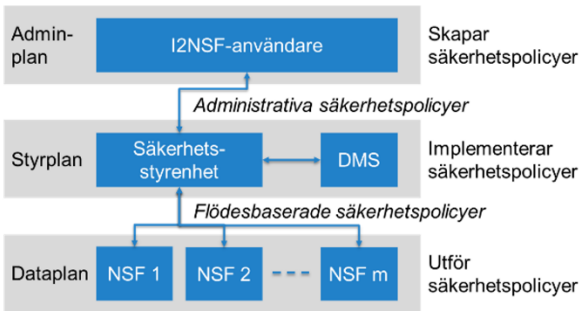
Arbetsgruppens uppdrag presenteras i mer detalj på https://datatracker.ietf.org/wg/i2nsf/about/ [läst 2020-10-25].

[37] Författarnas översättning och förtydligande numrering. Originalets lydelse är ”A Network Security Function (NSF) is a function used to ensure integrity, confidentiality, or availability of network communications, to detect unwanted network activity, or to block or at least mitigate the effects of unwanted activity.” Se https://datatracker.ietf.org/wg/i2nsf/about/ [läst 2020-05-04].

[38] https://www.clavister.com/

[39] https://www.checkpoint.com/

Hanteringen av nätverkssäkerhetsfunktionernas säkerhetspolicyer följer i princip samma struktur som för hanteringen av förmedlingsfunktionernas policyer. Detta innebär att uppdelningen i administrationsplan, styrplan och dataplan är tillämpbara även för säkerhetsfunktionerna. Figur 12 visar den övergripande arkitekturen från I2NSF relaterat till de tre funktionella planen för nätverks-säkerhetsfunktioner (Jeong m.fl., 2019). Den principiella indelningen med ansvarsfördelning för att skapa, implementera och tillämpa policyer är identisk som för förmedlingsfunktionerna, med skillnaden att det är säkerhetspolicyer snarare än flödespolicyer som kommuniceras mellan lagren.




Figur 12. I2NSF-modellens tre funktionsplan för nätverkssäkerhetsfunktioner. DMS står för Developer's Management System och är hjälpfunktioner för driftsättning av NSF:er.

Genom en uppdelning i tre plan med väldefinierade gränssnitt – framför allt mellan styr- och dataplan – siktar I2NSF på att underlätta integration och interoperabilitet mellan nätverkssäkerhetsfunktioner från olika leverantörer. Exempelvis är målet att det ska gå att hantera nätverkssäkerhetsfunktioner med olika funktion och förmågor i distribuerade och dynamiska nätverksmiljöer (Hares m.fl., 2017).

På administrationsplanet i figur 12 återfinns *I2NSF-användare*. Dessa är ansvariga för att skapa säkerhetspolicyer och kan exempelvis vara integrerade system för nätverkshantering och nätverksövervakning (Lopez m.fl., 2018).

Styrplanet ansvarar för att implementera säkerhetspolicyerna. Förutom säkerhets-styrenheterna finns även en extra funktionsgrupp som betecknas *Developer's Management Systems* (DMS) på styrplanet i I2NSF-modellen. De senare utgör hjälpfunktioner för att driftsätta nätverkssäkerhetsfunktioner (Jeong m.fl., 2019).

På dataplanet återfinns de enskilda nätverkssäkerhetsfunktionerna som är utplacerade i nätverket.

Gränsytorna mellan planen utgår från de specifika egenskaper som säkerhets-policyerna har, vilket innebär att andra protokoll behöver nyttjas än de som används för flödespolicyer.

# 5 Protokoll

För att mjukvarudefinierade nätverk och nätverksvirtualisering ska vara praktiskt genomförbart krävs ett antal olika nätverksprotokoll för kommunikationen mellan olika funktioner i nätverk. Vissa protokoll, såsom Ethernet, IP, TCP och UDP, utgör generella basprotokoll medan andra är mer specifika för de typer av problem som nätverksvirtualisering och mjukvarudefinierade nätverk är tänkta att lösa. De senare protokollen används exempelvis för att separera trafikflöden i nätverk och för att kommunicera mellan de olika funktionella planen i ett mjukvarudefinierat nätverk.

Protokoll för att separera flöden inkluderar bland annat *VLAN*, *QinQ* och *VXLAN*. Protokoll för kommunikation mellan de funktionella planen inkluderar exempelvis *Forces, Openflow* och *Opflex*. Följande avsnitt ger kortfattade introduktioner till dessa protokoll.

## 5.1 Protokoll för separation av trafikflöden

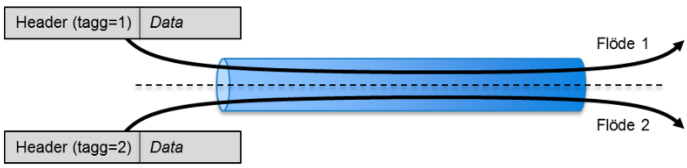
Det finns många användningsfall där det är önskvärt att separera olika trafikflöden i samma fysiska nätverk. Det kan exempelvis handla om att separera trafiken som hör till olika kunder hos en infrastrukturleverantör eller att skilja olika delar av ett företags interna nätverkstrafik från varandra. Med framväxten av nätverksvirtualisering och mjukvarudefinierade nätverk har dessa behov ökat allt mer, speciellt där flera system eller kunder delar på en och samma fysiska miljö.

VLAN och QinQ[40] är två näraliggande utökningar till Ethernet som ger möjlighet till virtuella lokala nätverk (eng. virtual LAN, VLAN) och är därmed protokoll på länklagret. Protokollen märker varje paket med så kallade *taggar* som utgör unika identifierare för de virtuella nätverken. I VLAN används en tagg om 12 bitar, vilket ger möjlighet att särskilja 4094 flöden[41] i trafiken. QinQ bygger på ett slags dubbel VLAN-taggning, vilket innebär att två taggar om 12 bitar vardera används. QinQ ger därmed en teoretisk möjlighet att särskilja knappt 16,8 miljoner olika flöden.

Taggningen innebär att flödena hålls separerade i nätverket på ett sätt som emulerar att de olika trafikflödena gick i fysiskt separata nätverk som illustreras i figur 13.

[40] VLAN och QinQ är specificerade i standarden IEEE 802.1Q-2018 (IEEE, 2018). I standarden kallas taggen i VLAN för *customer* VLAN (C-VLAN) medan den tillkommande taggen i QinQ kallas för *service* VLAN (S-VLAN).

[41] 12 bitar ger 4096 olika kombinationer, men två av dessa är reserverade för specifika användningar.



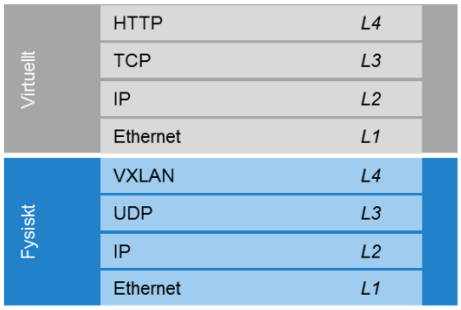

Figur 13. Paket uppdelade i separerade flöden med hjälp av taggning.

Till skillnad från VLAN och QinQ så är VXLAN ett applikationsprotokoll på internetmodellens fjärde lager. VXLAN inkluderar en tagg om 24 bitar, vilket i teorin ger samma antal möjliga flöden som för QinQ. Då VXLAN är ett applikationsprotokoll används underliggande protokoll i form av UDP och IP som i sin tur transporteras över exempelvis Ethernet. I VXLAN-paketen bäddas sedan det virtuella nätverkets trafik in, vilket ger ett slags duplicerad protokollstruktur.

En effekt av att VXLAN är ett applikationslagerprotokoll är att trafiken inte begränsas till inom ett lokalt nätverk. VXLAN-trafik kan skickas över längre distanser och över internet, vilket exempelvis kan underlätta nätverks-virtualisering som omfattar geografiskt åtskilda datacenter.

Figur 14 visar ett exempel på hur VXLAN används. I exemplet transporteras webbtrafik via protokollet HTTP över ett virtuellt nätverk där VXLAN används som bärare för den virtuella nätverkstrafiken. De fyra understa lagren i bilden, det vill säga Ethernet, IP, UDP och VXLAN, hanteras direkt av det fysiska nätverket och samverkar för att transportera trafik på det virtuella nätverket. De fyra översta lagren, det vill säga Ethernet, IP, TCP och HTTP, utgör trafik i det virtuella nätverket. VXLAN *inkapslar* (eng. encapsulates) därmed det virtuella nätverkets trafik, vilket i praktiken innebär att protokollstackens fyra lager upprepas – först i det fysiska nätverket och sedan i det virtuella.

Inkapslingen gör att det virtuella nätverket upplevs som ett fysiskt nätverk med en normal protokollstack, så när som på att det virtuella länklagret av naturliga skäl inte inkluderar det fysiska mediet. Detta emuleras i stället genom det fysiska nätverkets applikationslager, vilket i detta exempel utgörs av VXLAN-protokollet.




Figur 14. Protokollhierarki för att hämta en webbsida (via HTTP) över ett virtuellt nätverk som skapats med VXLAN.

En aspekt med protokoll såsom VLAN, QinQ och VXLAN är dessa inte inkluderar några specifika säkerhetsmekanismer. Det finns därmed inga hinder för någon entitet i nätverket att injicera paket på ett virtuellt nätverk, att läsa paket i andra virtuella nätverk eller tagga om paket så att de byter virtuellt nätverk.

## 5.2 Protokoll mellan styr- och dataplan

De protokoll som används för att kommunicera mellan styrplan och dataplan är specifika för mjukvarudefinierade nätverk och används typiskt inte i andra sammanhang. Protokollen används för vanligtvis för att skicka policy-uppdateringar från styrenheten till nätverksfunktionerna och för att rapportera olika händelser och statusuppdateringar i motsatt riktning.

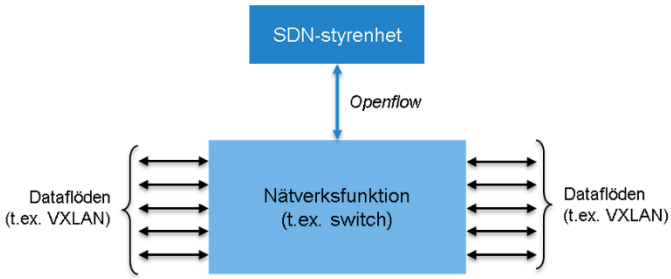
*Forwarding and Control Element Separation* (ForCES)[42] är ett öppet och fritt tillgängligt protokoll som specificerar hur kommunikationen ska ske mellan styrenheter och nätverksfunktioner. Forces bygger på att de kommunicerande enheterna i nätverket utrustas med så kallade element, vilka utgör väldefinierade implementationer av Forces-protokollet. Elementen definieras i två varianter beroende på deras placering: *styrelement* (eng. control element, CE) används i styrenheterna och *förmedlingselement* (eng. forwarding element, FE) används i nätverksfunktionerna.

Forces är inte uttryckligen avsett för mjukvarudefinierade nätverk[43] och kan även användas internt i en fysisk nätverksenhet, till exempel i större så kallade

[42] Specifikationen för ForCES är uppdelad i flera delstandarder. Se exempelvis Yang, L., Dantu, R., Anderson, T. & Gopal, R. (2004). *Forwarding and Control Element Separation (ForCES) Framework*, (RFC 3746, IETF) samt Doria, A., Hadi Salim, J., Haas, R., Khosravi, H., Wang, W., Dong, L., Gopal, R. & Halpern, J. (2010). *Forwarding and Control Element Separation (ForCES) Protocol Specification.* (RFC 5810, IETF).

[43] Termen ”mjukvarudefinierade nätverk” sägs ha myntats år 2009, vilket alltså är flera år efter publiceringen av första specifikationen för ForCES.

*bladswitchar* som byggs upp av flera separata insticksenheter för med olika ansvar för styrning och datavägar.




Figur 15. Openflow mellan styrenhet och nätverksfunktion.

Openflow[44] är ett öppet och fritt tillgängligt protokoll för kommunikation mellan styrenheter och nätverksfunktioner i SDN-lösningar. Figur 15 visar ett scenario med en styrenhet och en switch, där Openflow utgör bryggan mellan dessa. Styrenheten håller den övergripande nätverksbilden och förmedlar den information som switchen behöver genom *flödestabeller*. Dessa tabeller utgör ett slags regeluppsättningar för trafikflöden, där olika fält i inkommande paket undersöks utifrån flödestabellens innehåll och behandlas utifrån de regler som matchar. Exempelvis kan en regel undersöka destinationsadressen i Ethernet-ramen och utifrån den bestämma på vilken utgående port som paketet ska skickas vidare.

Openflow kan även överföra information om trafikflödena till styrenheten. Ett exempel är när ett paket kommer in till en switch och denna inte har någon flödestabellsregel som matchar paketet. Switchen kan då skicka vidare hela eller delar av paketet till styrenheten via Openflow, så att styrenheten kan uppdatera den övergripande nätverksbilden och ta beslut om hur ett sådant paket ska behandlas av switchen. Svaret från styrenheten blir typiskt en uppdatering av flödestabellerna, där regler lagts till för det tidigare omatchade paketet.

Ciscos (2014b) Opflex är ett proprietärt protokoll[45] som utgår från en så kallad deklarativ modell, till skillnad från Openflows imperativa modell. Den deklarativa modellen i Opflex överför policyer på applikations- eller flödesnivå, vilket lämnar en viss grad av beslutsfattande till dataplanet. Den imperativa modellen i Openflow betyder istället i princip att styrenheten i detalj beskriver hur respektive paket ska hanteras av dataplanet och att dataplanet därmed aldrig behöver ta några egna beslut om paketen.

[45] OpFlex har publicerats som ett utkast genom IETF men har inte röstats igenom. Tiden för omröstning gick ut 2017. Se https://datatracker.ietf.org/doc/draft-smith-opflex/ [läst 2020-10-25].

Med den deklarativa modellen i Opflex blir styrningen av nätverket distribuerad i viss utsträckning, vilket exempelvis kan ge bättre skalbarhet än styrning som baseras på den imperativa modellen i Openflow (Latif m.fl., 2020). Samtidigt ger Opflex mindre programmerbarhet i nätverket jämfört med Openflow (Latif m.fl., 2020).

Det är inte bara själva styrningen som påverkas av de olika protokollens egenskaper. Protokollen påverkar även hur styrenheterna samlar in information från nätverksfunktionerna, exempelvis angående nätverkets topologi och trafikflöden. När systemet använder Opflex görs i regel topologi-utforskningen med ett separat, Cisco-specifikt protokoll som heter *Cisco Discovery Protocol* (CDP). När systemet är uppbyggt med Openflow görs utforskningen i regel med *Openflow Discovery Protocol* (OFDP).

OFDP är inte standardiserat, men bygger på en kombination av funktioner i Openflow och det standardiserade protokollet *Link Layer Discovery Protocol* (LLDP) (Pakzad m.fl., 2016). LLDP är specifikt framtaget för att möjliggöra utforskning av nätverkstopologi genom att varje nätverksfunktion frågar samtliga grannar om deras kunskap om nätverkets topologi, för att sammanställa och delge denna information till grannarna (IEEE, 2016). När alla enheter gör detta kommer hela nätverkets topologi till slut att vara känd.

Nackdelen med LLDP är att all intelligens kring protokollet måste vara implementerad ute i nätverksfunktionerna. OFDP ändrar detta genom att nyttja själva upptäcktsmekanismen i LLDP – en specifik typ av meddelande i protokollet – tillsammans med Openflows mekanismer för att injicera och fånga upp paket i de styrda nätverksfunktionerna, så att intelligensen kan flyttas till styrenheterna. Genom OFDP kan styrenheterna injicera LLDP-meddelanden i valfria utgående nätverksanslutningar på nätverksfunktionerna för att sedan fånga upp dem i mottagande nätverksfunktioner. Därigenom kan styrenheterna kartlägga anslutningarna mellan nätverksfunktionerna och få en karta över nätverket.

# 6 Systemperspektiv

I detta kapitel sätts mjukvarudefinierade nätverk i ett systemperspektiv där egenskaperna relateras till rollen som komponent i ett komplett IT-system. Kapitlet ger en diskussion kring mjukvarudefinierade nätverk utifrån olika funktionella och administrativa perspektiv. Slutligen ges en diskussion kring de säkerhetsutmaningar som tillkommer vid användande av mjukvarudefinierade nätverk.

## 6.1 Beroenden

Nätverk utgör endast en del av ett komplett IT-system. Dessutom kräver nätverk i sig ofta stöd från IT-systemets övriga infrastruktur i form av olika gemensamma funktioner och tjänster. Detta gäller exempelvis virtualiseringsfunktioner, administrativa tjänster och funktioner för spårbarhet.

*Virtualiseringstjänster* skapar och hanterar den virtuella miljö som exempelvis kan inrymma virtuella nätverksfunktioner och det mjukvarudefinierade nätverkets styrenheter. Virtualiseringstjänsterna inkluderar bland annat:

- Administrationstjänster som används för att sköta och övervaka virtualiseringsmiljön.
- Utrullningstjänster (eng. provisioning services) som används för att driftsätta nya virtuella funktioner.
- Migrationstjänster som används för att flytta virtuella funktioner mellan olika fysiska maskiner.
- Lagringstjänster som används för att hantera den virtuella miljöns lagringsmedia och göra dem tillgängliga för respektive virtuell funktion.

*Systemtjänster* används brett i IT-system för att förenkla gemensam och synkroniserad styrning av systemet i fråga. Systemtjänster som används av nätverkets delar kan exempelvis inkludera:

- Övervaknings- och administrationstjänster används för att styra och övervaka systemets funktioner. Dessa tjänster använder protokoll såsom SNMP[46] och Netconf[47].

[46] *Simple Network Management Protocol* (SNMP) är ett protokoll som tillåter en klient (typiskt en övervaknings- och administrationsmjukvara) att kontakta olika entiteter i nätverket för att hämta status och göra inställningar. SNMP överför meddelanden som är kodade enligt ASN.1.

[47] *Network Configuration Protocol* (NETCONF) är en efterföljare till SNMP som är bättre anpassad för att göra inställningar. Informationen i NETCONF-meddelanden är kodad i XML.

- Tidstjänster används för att upprätthålla en gemensam tid i systemet, exempelvis via protokollet NTP[48].
- Autentiseringstjänster, såsom Cisco Identity Services Engine (ISE) och Microsoft Active Directory (AD), som används för att säkerställa en gemensam rättighetsdatabas för användare i systemet. Dessa tjänster kan exempelvis nyttja protokollen Radius, Kerberos och LDAP[49].
- Loggnings- och spårbarhetstjänster används för att samla in data om olika händelser i systemet. Dessa tjänster kan exempelvis nyttja protokollet Syslog[50].

Många av dessa tjänster är vitala för att mjukvarudefinierade nätverk och nätverksfunktioner ska fungera väl.

## 6.2 Nätverksadministration

Nätverksadministration innefattar många olika uppgifter, såsom konfiguration av delar av de tjänster som tas upp i avsnitt 6.1. Det kan göras manuellt men ofta nyttjas verktyg som exempelvis Puppet[51] för att centralisera och förenkla förändringar av konfigurationer och säkerställa att dessa är konsekventa.

Med mjukvarudefinierade nätverk ökar administratörens möjlighet till detaljstyrning över nätverkets trafikflöden genom olika policyer. Att nätverket blir mer programmerbart ökar komplexiteten men ger även bättre möjligheter till finmaskig kontroll av nätverkets flöden. Den centrala styrningen skapar en övergripande topologisk karta över nätverket som underlättar förståelsen för hur nätverket förändras utifrån konfigurationer och hur nätverket kan behöva förändras.

### 6.2.1 Konfigurera flöden

Genom att mjukvarudefinierade nätverk flyttar delar av besluten från förmedlingsfunktionerna till de centrala styrenheterna blir nätverkets flöden även mer dynamiskt programmerbara. Styrenheterna kan detaljstyra vidaresändnings-besluten i hela nätverket utifrån givna policyer från administratörer och applikationer. Som beskrivs i avsnitt 4.2 kan trafiken styras utifrån exempelvis vilka protokoll som paketet innehåller kombinerat med käll- och

[48] *Network Time Protocol* (NTP) används för att synkronisera klockor med relativt hög precision över nätverk.

[49] *Remote Authentication Dial-In User Service* (RADIUS), *Kerberos* och *Lightweight Directory Access Protocol* (LDAP) används för att autentisera entiteter mot en katalogserver.

[50] *Syslog* är ett protokoll för att överföra textbaserade loggmeddelanden med tillhörande metadata (exempelvis tidsstämplar) till en loggserver.

[51] https://puppet.com/

destinationsadresser för både Ethernet- och IP-lagren samt destinationens TCP-port. Förmedlingsfunktionerna kan utifrån styrningen därför agera som exempelvis router, switch, lastbalanserare eller brandvägg.

Segmentering av nätverk har länge nyttjats för att öka säkerheten och minska risken för att exempelvis intrång eller skadlig kod ska kunna sprida sig i nätverk. Med segmenteringen kan perimeterskydd skapas, där varje segment skyddas från trafiken i andra segment. För att segmentera nätverkstrafik mellan entiteter *inom* ett segment används mikrosegmentering[52], vilket ger en finkornig styrning av tillåtna trafikflöden och åtkomst mellan olika entiteter i systemen (Vmware, 2014).

Den finkorniga styrning som mikrosegmenteringen kräver kan åstadkommas genom mjukvarudefinierade nätverk och nätverksvirtualisering, men kan även åstadkommas i traditionella nätverk med en blandning av traditionella tekniker för segmentering (Rapp, 2019). Traditionella tekniker för segmentering bygger på en blandning av tekniker såsom brandväggar, VLAN-uppdelning av trafik och olika former av åtkomstkontroll (Oltsik, 2017). Mjukvarudefinierade nätverk gör det dock möjligt att bryta ner arbetsflöden, tjänster och applikationer i mikrosegment med en teknologi, vilket gör det mer praktiskt genomförbart och dessutom mer dynamiskt förändringsbart (Oltsik, 2017, Vmware, 2014). Eftersom mjukvarudefinierade nätverk och virtualisering gör det enklare att dynamiskt förändra ett nätverk efter behov, kan exempelvis administratörer skapa ett isolerat nätverkssegment för en grupp besökare, eller inkludera fler servrar i nätverket för en annan grupp för att tillfälligt utöka beräkningskraften.

### 6.2.2 Tjänstekedjor

I moderna nätverk finns det behov av tjänstekedjor som är flexibla nog att hantera såväl dynamisk efterfrågan från användare som dynamiska policyer i nätverket (Bhamare m.fl., 2016). Genom att det mjukvarudefinierade nätverkets styrenheter har såväl en bild av, som kontroll över trafikflödena i nätverket, har de därmed också möjligheten att dynamiskt styra trafiken genom tjänstekedjor utifrån nätverkets policyer.

Jämfört med traditionella nätverk kan mjukvarudefinierade nätverk underlätta nätverksadministratörers arbete med tjänstekedjor. Detta då potentiellt manuell konfiguration och trafikstyrning ersätts av nätverkspolicyer som översätts automatiskt till flödesregler av nätverkets styrenheter. Automatiken underlättar exempelvis när virtuella maskiner migrerar mellan olika platser i ett datacenter eller när användare flyttar i nätverket (Bhamare m.fl., 2016). Flöden kan då ledas

[52] Mikrosegmentering är en relativt ny term som först myntades av Vmware (Mämmelä, 2016).

om genom en likadan tjänstekedja anpassad för maskinens eller användares nya plats för att upprätthålla policyerna (Abujoda m.fl., 2016).

Med hjälp av automation kan tjänstekedjornas nätverksfunktioner dynamisk byta ordning baserat på händelser i nätverket och på vilken typ av trafik som färdas. Detta kan möjliggöra mer effektivt och mindre kostsamt användande av nätverksfunktioner (Shameli-Sendi m.fl., 2019).

En viktig aspekt av dynamiska tjänstekedjor är att såväl funktionsplacering som trafikflöden behöver optimeras dynamiskt för att upprätthålla effektiviteten, exempelvis genom nå lägsta belastning på systemet och tillräcklig tjänstekvalitet (Bhamare m.fl., 2016; Shameli-Sendi m.fl., 2019).

### 6.2.3 Topologi och monitorering

En central vy över topologi och trafikrörelser i ett nätverk ger information som bistår vid såväl nätverksadministration som analys av säkerhetsrelaterade händelser i systemen och nätverket. Ur ett säkerhetsperspektiv kan en central vy underlätta aktiviteter såsom hotjakt (eng. threat hunting), detektion av angrepp och hantering av pågående angrepp. Det öppnar möjligheten för att upptäcka anomalimönster utifrån hela nätverkets trafikrörelser. Mjukvarudefinierade nätverk ger inte mer information om nätverket än det går att få i traditionella nätverk, men tekniken kan underlätta att erhålla informationen. I traditionella nätverk behöver sensorer för inhämtning av trafikinformation installeras eller placeras på varje nätverksfunktion eller länk i nätverket. Detta kan vara svårt att realisera i stora nätverk menar Shin m.fl. (2016).

En topologisk karta ger en överblick över nätverket och hur det är kopplat. En topologisk karta kan innehålla bland annat nätverksfunktioner, klienter och servrar samt länkar däremellan (Hong m.fl., 2015). Mjukvarudefinierade nätverk skapades inte för att övervaka nätverk, utan endast för att styra förflyttningen av paket (Silva m.fl., 2017). Dock är upprätthållande av en korrekt topologisk karta fundamental för att styrningen i ett mjukvarudefinierat nätverk ska fungera (Hong m.fl., 2015). I ett traditionellt nätverk kan information om ett systems topologi erhållas genom att exempelvis samla in data genom protokollen LLDP och SNMP från alla enheter i nätverket. I mjukvarudefinierade nätverk som bygger på Openflow finns det vanligtvis inbyggd funktionalitet i styrenheten som inhämtar information om nätverkets topologi (se avsnitt 5.2 för detaljer).

För att styrenheterna ska kunna fatta informerade beslut exempelvis angående lastbalansering, krävs det att de har tillgång till relativt detaljerad information om trafikflödena i nätverket. Information om trafikflödet kan bland annat inkludera antal paket, paketens storlek och vilken VXLAN-tagg som används (Claise m. fl., 2013;Vmware, 2019). Informationen kan exempelvis användas för att detektera, diagnostisera och åtgärda problem i ett nätverk (Sflow, n.d.).

Ytterligare användningsområden är exempelvis att upptäcka tillgänglighetsattacker(DoS) eller för att optimera trafikflödet.

Grundläggande information om trafikflöden kan exempelvis fås genom SNMP, men mer detaljerad informationsinsamling kräver specialiserade protokoll såsom IPFIX, Netflow och Sflow. Samma protokoll kan användas både i traditionella och i mjukvarudefinierade nätverk. Det finns vissa anpassningar av protokollen som är specifikt riktade till mjukvarudefinierade nätverk. En särskild version av Sflow har anpassats för Openflow och ger styrenheten ytterligare översikt för beslut (da Silva m.fl., 2017). Flexam är en utökning av Openflow som ger viss information på paketnivå per flöde till styrenheten (Shirali-Shahreza & Ganjali, 2013). Inga av dessa protokoll ger dock fullständig paketinformation för djupare analyser, då detta kräver att en övervakningspunkt skapas som skickar insamlad trafikdata till ett övervakningssystem (Hedlund, 2013; da Silva m.fl., 2017). Forskning pågår för att bättre nyttja de mjukvarudefinierade nätverkens egenskaper för trafikövervakning (Queiroza m.fl., 2019).

## 6.3 Skalbarhet

I mindre nätverk räcker det i regel med en styrenhet för hela nätverket. När nätverket blir större kan en ensam styrenhet blir en flaskhals, vilket innebär att nätverket kan kräva flera samverkande styrenheter för att upprätthålla tillförlitlighet, tillgänglighet och prestanda.

Flera distribuerade styrenheter kan ge vissa fördelar men det leder också till utmaningar. Flera styrenheter bidrar till nätverkets tillförlitlighet och skalbarhet (Jain m.fl., 2019, s. 17; Kreutz m.fl., 2015, s. 54). Exempelvis menar Jain m.fl. (2019) att flera styrenheter kan hantera fel bättre om nätverkskomponenter på godtycklig plats i nätverket slutar fungera. Styrenheternas placering är ett eget forskningsområde, vanligtvis kallat *Controller Placement Problem* (CPP), som behandlar hur många styrenheter som behövs i nätverket och var de bör placeras. I regel är styrenhetens placering ett utbyte mellan prestanda och tillförlitlighet (Hock m.fl., 2013).

Hantering av vidareförmedlingsregler i nätverkskomponenter kan också bli en flaskhals i olika situationer. Exempelvis skickar vissa protokoll det första paketet i ett flöde till styrenheten och inväntar därefter en ny vidareförmedlingsregel från styrenheten. När många nya flöden initieras i ett nätverk kan därmed många nya vidareförmedlingsregler skapas på kort tid. Det kan påverka prestandan i nätverket om mycket styrtrafik skickas mellan nätverkskomponenter och styrenheten (Akyildiz m.fl., 2014). Nätverkskomponenternas begränsade minnesmängd för vidareförmedlingsregler kan också påverka nätverkets skalbarhet, exempelvis om många tjänstekedjor skapas. Då installeras många vidareförmedlingsregler i nätverkskomponenten för olika tjänstekedjor, vilket kan fylla upp nätverkskomponentens minne (Guo m.fl., 2016).

## 6.4 Interoperabilitet

Kreutz m.fl. (2015) menar att mjukvarudefinierade nätverk kan vara lösningen på att traditionella nätverk är komplexa och svårhanterliga. En nackdel i traditionella nätverk är att data- och styrplanet i regel är integrerade i samma hårdvarukomponent, vilket begränsar möjligheten att dynamiskt styra nätverksfunktionernas beteende under drift. En annan nackdel är att nätverksfunktioner från olika leverantörer hanteras på helt olika sätt, det vill säga att olika instruktioner eller protokoll används för att styra dem. Genom öppna protokoll och gränssnittsspecifikationer kan mjukvarudefinierade nätverk underlätta interoperabilitet så att komponenter från olika leverantörer kan användas tillsammans på ett effektivt sätt (Jain m.fl., 2019, s. 10; Kreutz m.fl., 2015).

Det kan dock uppstå problem när olika leverantörer implementerar gemensamma protokoll i sina nätverkskomponenter, exempelvis kan protokollspecifikationerna tolkas olika, vilket kan leda till att nätverkskomponenter beter sig olika (Kuzniar m.fl., 2012).

Inom produktområdet som inkluderar mjukvarudefinierade nätverk finns en tydlig splittring mellan olika leverantörer, där de valt olika lösningar och protokoll. Inom forskningsvärlden och öppen källkod är Openflow fortfarande stort (Latif, 2020). Cisco har däremot lämnat Openflow till förmån för det egna protokollet Opflex i produktsviten Application Centric Infrastructure (ACI) (Odom, 2020) Juniper nyttjar det generiska protokollet XMPP[53] i produkten Contrail Network (Juniper 2020). Genom denna splittring följer en påtaglig risk för inlåsningseffekter som står i stark kontrast med den ursprungliga tanken att mjukvarudefinierade nätverk skulle ge ökad interoperabilitet i nätverken.

Akyildiz m.fl. (2014, s.20) beskriver att mjukvarudefinierade nätverk behöver övervaka nätverket och samla information om exempelvis länkstatus. Ett problem är att övervakningskomponenter från leverantörer hanteras olika beroende på leverantör. Därför används ofta traditionella nätverksfunktioner för att övervaka nätverket trots att mjukvarudefinierade nätverk används.

## 6.5 Säkerhetsutmaningar

Mjukvarudefinierade nätverk introducerar nya säkerhetsutmaningar utöver de som finns i traditionella nätverk. Det gäller utmaningar såväl i att motstå antagonistiska angrepp som i att hantera oavsiktliga misstag.

Den centraliserade arkitekturen i mjukvarudefinierade nätverk medför att misstag kan ge utbredda effekter i systemen. En felaktig konfiguration av en central

[53] Extensible Messaging and Presence Protocol (XMPP) är ett protokoll som ursprungligen togs fram för att överföra chattmeddelanden, men som senare kommit att användas för att kapsla in alla möjliga meddelanden som kodats i XML. XMPP definieras i RFC 6120 (Saint-Andre, 2011).

funktion eller en felaktig nätverkspolicy kan få omfattande påverkan på nätverkets funktion och därmed på dess tillgänglighet.

Centraliseringen ger även nya möjligheter för angrepp på systemet, varför kritiska funktioner och gränssnitt behöver skyddas. Fortsättningen av det här avsnittet bygger på den analys som genomförts av Zhu m.fl. (2017).

Vad gäller applikationsplanet ökar exponering mot skadliga applikationer som kan påverka styrplanet och därigenom hela nätverket. Skadliga applikationer kan exempelvis leda till att information i nätverket läcker ut eller att nätverket slutar fungera. Det blir därför viktigt att validera applikationer innan de tas i drift och att implementera åtkomstkontroll vid styrenhetens norra gränssnitt så att endast betrodda applikationer och användare kan kommunicera med en styrenhet.

Styrenheterna är särskilt attraktiva mål för antagonister eftersom de kan såväl styra nätverket som hämta information om det. Det är centralt för säkerheten att skydda styrenheterna i styrplanet, både vad gäller tillgänglighet och riktighet. Följande lista ger några exempel på möjliga oönskade händelser:

- Styrenheterna sätts ur drift genom fysisk eller logisk påverkan vilket gör att nätverket slutar fungera.
- En eller flera styrenheter tas över av en angripare, som då kan kontrollera förmedlingen av alla flöden i nätverket.

I dataplanet är förmedlingsfunktionerna mål för att sabotera nätverket, försöka ta över styrenheten eller förändra trafikflöden. I följande lista ges några konkreta exempel på möjliga oönskade händelser.

- En switch som tagits över av angriparen skickar många förfrågningar till styrenheten som då överbelastas.
- Felaktiga flödestabeller installeras för att sabotera nätverkets funktionalitet.

Att upprätthålla en korrekt topologisk karta över nätverket är fundamentalt för att styrningen i nätverket ska kunna fungera korrekt. Om en angripare lyckas manipulera den topologiska kartan kan exempelvis trafiken ledas fel eller tillgängligheten i nätverket påverkas (Hong m.fl. 2015).

System baserade på mjukvarudefinierade nätverk och virtuella nätverks-funktioner är inte alltid homogena utan består av både fysiska och virtuella enheter och nätverksfunktioner. Zhu m.fl (2017) menar att det därför kan bli en komplex uppgift att säkra miljön. Istället för en traditionell statisk nätverkstopologi kan miljön vara dynamisk och förändras utifrån behov och belastning.

# 7 Diskussion

Detta kapitel diskuterar olika aspekter och vinklar på mjukvarudefinierade nätverk och de närliggande komponenter och tekniker som beskrivits tidigare i rapporten. Diskussionerna utgår från tre olika perspektiv: tekniken, säkerheten och Försvarsmakten.

## 7.1 Teknikperspektivet

Mjukvarudefinierade nätverk har blivit en mogen och allmänt accepterad teknik som används i många av dagens större IT-system.[54] De större leverantörerna inom datacenterlösningar har redan etablerade produkter för mjukvarudefinierade nätverk, exempelvis Vmware NSX[55], Cisco ACI[56] och Juniper Contrail[57]. Därtill finns ett antal lösningar med öppen källkod, såsom Open Network Operating System (ONOS)[58] och Opendaylight[59]. Alla dessa lösningar varierar vad gäller komplexitet, vilka komponenter som ingår och hur tätt de är integrerade med andra produkter. Vmware utgör ett exempel på den tätt integrerade änden av skalan, där NSX ingår som en del i deras produktsvit för det *mjukvarudefinierade datacentret* (eng. software-defined data center, SDDC).[60] I den mindre integrerade änden av skalan återfinns exempelvis ONOS som är en renodlad styrenhetslösning för mjukvarudefinierade nätverk. Ett system som baseras på ONOS måste därför bestå av nätverksfunktioner som kommer från andra leverantörer, men som nyttjar något styrprotokoll som även stöds av ONOS.

Idag pågår en fortsatt utveckling av mjukvarudefinierade nätverk som bland utmynnat i så kallade *avsiktsbaserade nätverk* (eng. intent-based networking).

---

[54] Exempelvis ska International Data Corporation (IDC) i sin marknadsanalys för perioden 2018–2022 ha poängterat att mjukvarudefinierade nätverk har passerat stadiet med ”breathless hype and fevered expectations” och befinner sig numera i en hälsosam tillväxtfas (Cooney, 2019).

[55] Vmware har en mjukvarulösning för datacenter inklusive virtuella nätverksfunktioner, se ”VMware NSX Data Center” (https://www.vmware.com/products/nsx.html). Systemet måste dock ha en fysisk nätverksinfrastruktur som exempelvis kan bestå av ett Cisco ACI-baserat nätverk (Fairfield 2020).

[56] ”Cisco ACI for Data Center”, Cisco Systems Inc., https://www.cisco.com/c/en/us/solutions/data-center-virtualization/application-centric-infrastructure/index.html [läst 2020-10-19].

[57] ”Contrail – SDN-enabled management and control software for simplified service delivery”, Juniper Networks, https://www.juniper.net/us/en/products-services/sdn/contrail/ [läst 2020-10-19].

[58] ”ONOS – Open Network Operating System”, Open Networking Foundation, https://www.opennetworking.org/onos/ [läst 2020-10-19].

[59] ”OpenDaylight + Open Networking”, OpenDaylight Foundation, https://www.opendaylight.org/ [läst 2020-10-19]

[60] I VMwares tappning bygger ”The Software-Defined Data Center” på flera VMware-produkter utöver NSX, såsom Cloud Foundation, vSAN, vRealize Suite och vCloud Suite. https://www.vmware.com/solutions/software-defined-datacenter.html [läst 2020-10-19].

Termen myntades år 2017 i ett blogginlägg från analyshuset Gartner (Lerner, 2017) och har sedermera spridit sig bland de större nätverksleverantörerna. Cisco lyfter fram avsiktsbaserade nätverk som nästa stora händelse inom automation av nätverk (Cisco, 2019). Där mjukvarudefinierade nätverk handlade om samordning och automation av policyer och konfigurationer, inkluderar avsiktsbaserade nätverk ett större grepp om automation av nätverksadministrationen. I avsiktsbaserade nätverk automatiseras även översättning från avsikt till policy och uppföljning av att nätverken beter sig korrekt (Cisco, 2019).

Numera är mjukvarudefinierade nätverk att betrakta som en mogen teknik med åtskilliga välanvända implementationer från olika tillverkare och ett omfattande marknadsgenomslag.[61] Allied Market Research (2020) visar att marknads-volymen för produkter som implementerar mjukvarudefinierade nätverk ligger på cirka 10 miljarder USD år 2020 och att den förväntas växa kraftigt de närmaste åren.

Med mognaden har mjukvarudefinierade nätverk kommit in som en naturlig del i systemlösningen, framför allt i mer komplexa system såsom datacenter och större företagsnätverk. Denna utveckling kommer sannolikt att fortsätta, vilket innebär att Försvarsmakten behöver förhålla sig till mjukvarudefinierade nätverk som ett allt vanligare faktum i kommersiella systemlösningar.

Även om mjukvarudefinierade nätverk idag torde vara vanligare i sammanhang med större IT-system, såsom datacenterlösningar, än i mindre system såsom kontorsnätverk, så är de användbara även i de mindre systemen. Mjukvarudefinierade nätverk kan exempelvis underlätta administrationen och öka flexibiliteten, speciellt i sammanhang där systemen förändras ofta. Där tidigare nätverkslösningar krävde att administratörerna konfigurerade alla nätverksfunktioner individuellt utifrån nätverkspolicyerna, kan mjukvaru-definierade nätverk göra så att detta sker automatiskt utifrån policyer som definierats på en högre abstraktionsnivå. Med centrala styrenheter som aggregerar information från nätverket möjliggörs nya applikationer för att skapa och hantera nätverkspolicyer på hög nivå.

I traditionella nätverk används nätverksfunktioner med relativt statisk funktion som på sin höjd går att förändra genom de konfigurationsmöjligheter som lagts in av tillverkaren. Med mjukvarudefinierade nätverk öppnas möjligheten att lägga

[61] Mognadsgraden kan lättast exemplifieras genom den uppsjö av olika produkter som finns på marknaden idag. Produkter som implementerar mjukvarudefinierade nätverk inkluderar exempelvis Arista Software Driven Cloud Networking (https://www.arista.com/en/products/software-driven-cloud-networking), Ciena MCP (https://www.ciena.com/insights/what-is/What-is-MCP.html), Cisco ACI (https://www.cisco.com/c/en/us/solutions/data-center-virtualization/application-centric-infrastructure/index.html) , Juniper Contrail (https://www.juniper.net/uk/en/products-services/sdn/contrail/), ONOS (https://opennetworking.org/onos/) , Opendaylight (https://opennetworking.org/onos/) och Vmware NSX (https://www.vmware.com/products/nsx.html).

in nya funktioner på styrplanet och administrationsplanet som påverkar nätverksfunktionernas funktion. Detta kan exempelvis ge utvecklare möjlighet att genom policyers utformning kombinera olika nätverksfunktioner för att skapa funktionalitet som är större än summan av komponenterna. I och med att detta kan ske utan att modifiera respektive nätverksfunktion kan tröskeln för att skapa nya nätverksfunktioner eventuellt bli lägre (Jain et al., 2019; Raj & Raman, 2019).

När mjukvarudefinierade nätverk används tillsammans med andra tekniker för virtualisering – såsom virtualisering av nätverksfunktioner, virtualisering av datorer samt nätverksvirtualisering – blir en effekt att kopplingen mellan systemets funktion och den underliggande hårdvarans suddas ut. I och med detta har kombinationen av mjukvarudefinierade nätverk och virtualisering av nätverksfunktioner potential att ge nätverk som är flexibla och anpassningsbara. Samtidigt finns en risk för att administratörerna förlorar sin översikt och förståelse för hur nätverket ser ut och fungerar, vilket kan leda till oväntade effekter när förändringar sker i systemet. Vi tar upp ett par exempel på sådana oväntade följdverkningar i avsnitt 7.2, säkerhetsperspektivet.

En viktig insikt som ofta tycks försvinna i diskussioner om virtualisering och mjukvarudefiniering är att förändringarna i regel innebär att ytterligare lager av mjukvara läggs till i systemet utan att systemet nödvändigtvis tillförs funktioner för slutanvändarna. Den underliggande hårdvaran och de produktiva funktioner som systemet tillhandahåller är i grund och botten desamma. Dessa delar kan förvisso förändras till viss del, men de försvinner inte ur systemen.

Eftersom mjukvarudefinierade nätverk huvudsakligen handlar om nätverkets styrplan finns inga egentliga hinder för att blanda traditionella, integrerade nätverksfunktioner med sådana som anpassats för mjukvarudefinierade nätverk. Det kan exempelvis ske i en övergångsfas när ett befintligt nätverk ska uppgraderas, men det kan även vara ett aktivt val att blanda teknikerna. Sådana blandade nätverk kallas för *mjukvarudefinierade hybridnätverk* (eng. hybrid SDN) och får olika egenskaper beroende på hur teknikerna blandas (Sandhya et al., 2017).

Med öppna och standardiserade gränssnitt för det sydgående gränssnittet mellan styrenheter och nätverksfunktioner skapas möjligheten att integrera produkter från olika leverantörer i ett heterogent nätverk som trots detta styrs med gemensamma policyer. I praktiken fungerar dock inte detta då det finns en konkurrens mellan flera (öppna och leverantörsspecifika) standarder där olika leverantörer har gjort olika vägval. Som exempel använde Cisco tidigare standarden Openflow för kommunikationen, men har på senare år fokuserat allt mer på sitt egna protokoll Opflex som är integrerat i produktsviten Cisco ACI.[62]

[62] Under 2010 lanserade Cisco Open SDN Controller (OSC), en Openflow-baserad styrenhet som utgick från Opendaylight. Några år senare lanserades produktsviten Cisco ACI som baseras på Ciscos egna

Denna fragmentering av marknaden där olika leverantörer följer olika standarder riskerar att leda till inlåsningseffekter, då IT-systemen endast kan kompletteras med utrustning från ett litet urval av leverantörer.

Även när samma standard implementeras av olika leverantörer finns det risk för skillnader i tolkning av standarden och hur mycket av den som implementeras. Om detta innebär att styrningen bara kan göras på den nivå som bestäms av ett slags minsta gemensamma nämnare av implementationerna hos de ingående komponenterna, kan effekten av det mjukvarudefinierade nätverket bli urvattnad varpå den tillförda komplexiteten och kostnaden inte är motiverad.

Såväl mjukvarudefinierade nätverk som virtualiserade nätverksfunktioner är etablerade tekniker på marknaden. Samtidigt utvecklas området fortfarande med trender inom både teknik och marknadsstruktur. En tekniktrend kan vara dragningen mot deklarativ styrning i mjukvarudefinierade nätverk och utvidgningen till avsiktsbaserade nätverk, såsom visas av Ciscos satsningar (Cisco, 2019). En annan tekniktrend kan vara fortsatt satsning på SD-WAN, drivet av behovet att ha flexibla och enkelt skalbara system som kan anpassas efter rådande situation som exemplifieras av de stora ändringarna i trafikmönster och trafikmängd i operatörernas nätverk när många började arbeta på distans som följd av COVID-19 (Edwards, 2020).

En marknadsdriven trend kan vara utbredningen av network slicing – där även tjänstekvalitet kommer in som en parameter i styrningen av nätverket – som följer av utvecklingen inom mobiltelefoniområdet. Tekniker och principer från network slicing kan mycket väl ta sig in i mjukvarudefinierade nätverk. Network slicing är fortfarande ett relativt nytt område, där det förutspås en kraftig tillväxt – en beräknad årlig tillväxt på över 20 % under de kommande fem åren – samtidigt som marknaden är fragmenterad och saknar tydliga nyckelaktörer (Mordor Intelligence, u.å.).

Under våren år 2020 stärkte teknikföretaget Nvidia, troligen mest känt för sina grafikprocessorer, sin position på nätverksmarknaden genom uppköp av företagen Cumulus Networks[63] och Mellanox[64]. Genom uppköp och partnerskap verkar Nvidia sikta på att bli en ledande aktör inom mjukvarudefinierade nätverk som baseras på öppna protokoll (Kerravala, 2020; Biebelhausen 2020), vilket kan påverka marknaden genom att interoperabilitet mellan leverantörer kommer mer i fokus.

protokoll Opflex (Odom, 2020). OSC slutade säljas under 2017 och sista supportdatum passerades under våren 2020 (Cisco, 2016).

[63] https://blogs.nvidia.com/blog/2020/05/04/nvidia-acquires-cumulus/

[64] https://nvidianews.nvidia.com/news/nvidia-completes-acquisition-of-mellanox-creating-major-force-driving-next-gen-data-centers

Fortsatt utveckling inom mjukvarudefinierade nätverk och virtualiserade nätverksfunktioner tillsammans med en öppnare marknad kan förhoppningsvis leda till att de ursprungliga visionerna om exempelvis leverantörsoberoende, effektivitet och skalbarhet kan uppfyllas i högre grad än idag.

## 7.2 Säkerhetsperspektivet

Som vi sett exempel på i avsnitt 6.2, kan mjukvarudefinierade nätverk möjliggöra förbättrade säkerhetslösningar i nätverken. Detta går att åstadkomma främst genom den centraliserade information som styrenheterna har över nätverket och aktiviteterna i detta samt den kontroll som styrenheterna kan utöva i nätverket. Informationen kan användas för övervakning och uppföljning, till exempel för att hitta okända enheter eller för att hitta oönskad aktivitet. Kontrollen ger möjlighet att styra nätverket mer finmaskigt och händelsestyrt, där exempelvis mikrosegmentering kan underlätta nolltillitslösningar[65] (eng. zero trust) och dynamisk omstrukturering av trafik kan underlätta isolering av oönskade händelser eller motverka överbelastning.

Samtidigt måste systemägare och utvecklare ta hänsyn till systemens komplexitet som mycket väl kan öka med mjukvarudefinierade nätverk. Tillförlitligheten i systemen kan påverkas negativt när komplexa interaktioner i systemen kan ändra beteendet på stor skala utan att utvecklarna och administratörerna kan förutsäga konsekvenserna. Exempelvis som när en dåligt utformad brandväggsregel överbelastade stora delar av CDN[66]-tjänsten Cloudflare (Graham-Cumming, 2019) eller vid driftavbrottet i Amazons datacentertjänst EC2 som inträffade på grund av en felaktig konfiguration av en router (AWS Team, 2011). Även om inget av dessa exempel direkt handlar om mjukvarudefinierade nätverk så illustrerar de en viktig aspekt av moderna IT-system: komplexiteten är så hög att det är omöjligt att på förhand kunna eliminera alla potentiella fel. Denna typ av ”normala olyckor” – ett uttryck som myntades av Perrow (1984) i boken *Normal Accidents* – kan förväntas i system med tillräckligt hög komplexitet som inte har mycket omfattande redundans i rutiner och säkerhetsmekanismer.

Utöver normala olyckor tillkommer de sårbarheter som aktivt kan utnyttjas för att påverka eller få tillgång till systemen i antagonistiskt syfte. Varje nytt lager av komplexitet som tillförs i ett system innebär automatiskt en större angreppsyta med exempelvis fler rader kod i mjukvara som kan innehålla sårbarheter.

[65] Zero trust innebär att system byggs utifrån principerna att det alltid finns hot i nätverket, oavsett om de är externa eller interna. För att uppnå nolltillitens principer behöver varje enhet, användare och nätverksflöde kontrolleras.

[66] Content Delivery Network

Samtidigt har mjukvarudefinierade nätverk och vidareutvecklingen avsiktsbaserade nätverk potential att möjliggöra nya, innovativa nätverkssäkerhetsfunktioner. Det pågår omfattande forskning inom området, exempelvis inom detektion av angrepp med hjälp av maskininlärning och så kallade *tensornätverk* (Wanga m.fl., 2020), finmaskigare åtkomstkontroll (Aschoff m.fl., 2017), och *Moving Target Defence* (MTD) genom slumpmässiga virtuella IP-adresser (Scott-Hayward m.fl., 2013). Dessutom finns ett antal forskningsinitiativ inom autonom säkerhet, exempelvis kring hur mjukvarudefinierade nätverk, virtualisering av nätverksfunktioner och maskininlärning kan samverka för att åstadkomma självkonfigurering, självläkning, självoptimering och självförsvar hos system (Benzaid & Taleb, 2020).

Med styrenheternas kunskap om nätverket och deras styrmöjligheter öppnas nya vägar i utvecklingen av nätverkssäkerhetsfunktioner, speciellt om de kombineras med information och funktioner hos andra nätverksadministrativa system. Mycket av forskningen är dock fortfarande i tidiga stadier, vilket sannolikt innebär att det kommer att ta ett antal år innan de potentiella säkerhetsfunktionerna återfinns i färdiga produkter.

Virtualiserade nätverkssäkerhetsfunktioner kan i placeras friare i systemet än motsvarande fysiska enheter då de även kan läggas in i systemets virtuella miljöer, exempelvis nära eller i systemets servrar. För att förenkla detta behöver de virtuella miljöerna standardiseras så att samma nätverkssäkerhetsfunktion kan implementeras i den virtuella miljön på exempelvis en fysisk nätverksenhet och en server. På grund av det starka beroendet till den virtuella miljöns gränsytor menar Zhu m.fl. (2017) att det kan bli svårt att hitta platser där ytterligare säkerhetsfunktioner kan placeras, varför systemen riskerar att bli låsta till det utbud av säkerhetsfunktioner som finns i respektive virtuell miljö.

Säkerhetsarbete är alltid en avvägning mellan intressen och funktioner som står i konflikt med varandra och därför måste fördelarna med mjukvarudefinierade nätverk vägas mot riskerna som tekniken för med sig. Denna avvägning går inte att göra på ett generellt plan, utan måste göras individuellt för varje system och situation.

## 7.3 Försvarsmaktsperspektivet

Som nämnts tidigare i diskussionen måste Försvarsmakten förhålla sig till mjukvarudefinierade nätverk då dessa har blivit en etablerad teknik. Försvarsmakten har många IT-system av vilt skild karaktär, från stora och komplexa datacenterlösningar till fristående datorer. Systemen hanterar information med olika säkerhetsimplikationer, från öppen och lågkritisk information till kvalificerat hemlig information samt system med mycket höga krav på tillgänglighet. Olika system har därmed mycket olika kravbild och är uppbyggda kring tekniska lösningar som skiljer sig åt ordentligt.

Tanken med mjukvarudefinierad styrning av nätverk är inte ny när det gäller system med högre säkerhetskrav. *Reaktiva nätverk* är ett koncept som togs fram av FOI på uppdrag av Försvarsmakten i början av 2010-talet (Gustafsson m.fl., 2012). Reaktiva nätverk fokuserar på logisk isolation av användartrafik genom att dynamiskt styra nätverksåtkomstsregler på användarnivå och klientnivå i nätverksenheterna. Till skillnad från reaktiva nätverk är det inte isolationen som är i fokus i mjukvarudefinierade nätverk. Dessa fokuserar i stället på styrning av trafikflöden utifrån nätverkspolicyer – en princip som förvisso kan nyttjas för att uppnå en viss nivå av logisk isolation.

I dagsläget är tekniken för mjukvarudefinierade nätverk främst inriktad på datacenterlösningar, större företagsnätverk och nätoperatörer. Detta kan mycket väl ändras på sikt så att mjukvarudefinierade nätverk blir allt vanligare även i mindre system. Då Försvarsmakten huvudsakligen bygger sina IT-system med standardkomponenter som köps på den kommersiella marknaden kan därmed mjukvarudefinierade nätverk bli en oundviklig komponent i vissa system.

Även om mjukvarudefinierade nätverk och andra mjukvarubaserade lösningar kanske inte är mogna för att uppfylla Försvarsmaktens krav på separation mellan olika system på högre säkerhetsnivåer så kan de ändå vara användbara komponenter i många system. Genomtänkt användning av mjukvarudefinierade nätverk och virtualiseringslösningar kan exempelvis underlätta driftsättning och administration av mer komplexa system, även om detta sker inom avgränsade zoner i systemet. Dynamiska nätverksmiljöer utgör ett specifikt fall där mjukvarudefinierade nätverk kan vara värdefulla då tekniken kan möjliggöra förenklad administration.

Innovation inom säkerhetslösningar är ett område som potentiellt kan skapa stort mervärde för Försvarsmakten, speciellt kopplat till tidig upptäckt av intrång och för att isolera händelser i nätverken. Många av dessa lösningar är dock fortfarande i forskningsstadiet och mycket arbete återstår innan de kan finnas i praktiskt bruk.

Som exempel på ett militärt användningsområde för mjukvarudefinierade nätverk finns ett Nato-förslag kring hur dessa kan förenkla sammanställning av cyberlägesbilden i taktiska koalitionsnätverk (Mishra m.fl., 2016). Förslaget utgår från att varje land i koalitionen har ett eget nätverk med egna styrenheter som ansvarar för respektive nätverk. Styrenheterna kopplas sedan samman genom speciella gränsöverskridande östvästliga gränssnitt, vilket gör att styrenheterna för respektive land kan utbyta information med varandra. Genom dessa gränssnitt kan styrenheterna exempelvis dela information om pågående händelser eller anomalier i nätverket, för att på så sätt underlätta den gemensamma lägesbedömningen (Mishra m.fl, 2016).

Som alltid när nya tekniska lösningar växer fram och integreras i systemen är det viktigt att vara medveten om teknikens begränsningar och hur systemen

påverkas, exempelvis när det gäller tillförlitlighet och IT-säkerhet. Denna medvetenhet är av naturliga skäl extra viktig i många av Försvarsmaktens IT-system då dessa utsätts för hot från avancerade motståndare.

# Referenser

Abujoda, A., Kouchaksaraei, H. R., & Papadimitriou, P. (2016). SDN-based source routing for scalable service chaining in datacenters. In Mamatas L., Matta I., Papadimitriou P., & Koucheryavy Y. (Eds), *International Conference on Wired/Wireless Internet Communication* (pp. 66–77). Springer. https://doi.org/10.1007/978-3-319-33936-8_6

Akyildiz, I. F., Lee, A., Wang, P., Luo, M., & Chou, W. (2014). A roadmap for traffic engineering in SDN-OpenFlow networks. *Computer Networks*, 71, 1–30. https://doi.org/10.1016/j.comnet.2014.06.002

Alharbi, T., Portmann, M., & Pakzad, F. (2015). The (in)security of topology discovery in software defined networks. In *2015 IEEE 40th Conference on Local Computer Networks (LCN)* (pp. 502–505). IEEE. https://doi.org/10.1109/LCN.2015.7366363

Allied Market Research (2020). *Software defined networking – Market – Opportunity and Forecast, 2020–2027*. https://www.alliedmarketresearch.com/software-defined-networking-market

Aschoff, R., Rosendo, D., Machado, M., Santos A. & Sadok, D. (2017) A Network Access Control solution combining OrBAC and SDN. In *2017 IFIP/IEEE Symposium on Integrated Network and Service Management*. IEEE. https://doi.org/10.23919/INM.2017.7987316

AWS Team (2011). *Summary of the Amazon EC2 and Amazon RDS Service Disruption in the US East Region*. Amazon. https://aws.amazon.com/message/65648/

Barakabitze, A. A., Ahmad, A., Mijumbi, R., & Hines, A. (2020). 5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges. *Computer Networks*, 167, 106984. https://doi.org/10.1016/j.comnet.2019.106984

Bednarz, A. (2018, 30 januari). What is microsegmentation? How getting granular improves network security. *Network world*. https://www.networkworld.com/article/3247672/what-is-microsegmentation-how-getting-granular-improves-network-security.html

Benzaid, C. & Taleb, T. (2020) ZSM Security: Threat Surface and Best Practices. *IEEE Network*, 34(3), 124–133. IEEE. https://doi.org/10.1109/MNET.001.1900273

Bhamare, D., Jain, R., Samaka, M., & Erbad, A. (2016). A survey on service function chaining. *Journal of Network and Computer Applications*, 75, 138–155. https://doi.org/10.1016/j.jnca.2016.09.001

Biebelhausen, J. (2020, 15 september). *Lenovo and NVIDIA Spark New Era of Open Networking*. Nvidia Blog. https://blogs.nvidia.com/blog/2020/09/15/lenovo-open-networking/

Braden, R. (red.) (1989). *Requirements for Internet Hosts -- Communication Layers* (RFC 1122). Internet Engineering Task Force (IETF). https://tools.ietf.org/html/rfc1122

Cisco (2014a). *Connecting Networks – Companion Guide*. Cisco Networking Academy.

Cisco (2014b). Cisco (2014). *OpFlex: An Open Policy Protocol White Paper.* https://www.cisco.com/c/en/us/solutions/collateral/data-center-virtualization/application-centric-infrastructure/white-paper-c11-731302.html

Cisco (2016). *End-of-Sale and End-of-Life Announcement for the Cisco Open SDN Controller 1.x.* https://www.cisco.com/c/en/us/products/collateral/cloud-systems-management/open-sdn-controller/eos-eol-notice-c51-738194.html

Cisco (2019). *2020 Global Networking Trends Report.* https://www.cisco.com/c/m/en_us/solutions/enterprise-networks/networking-report.html

Claise, B. Trammell, & P. Aitken (2013) *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information* (RFC 7011). Internet Engineering Task Force (IETF). https://tools.ietf.org/html/rfc7011

Cooney, M. (2019, 16 april). What is SDN and where software-defined networking is going. *Networkworld.* https://www.networkworld.com/article/3209131/what-sdn-is-and-where-its-going.html

da Silva, C.P., Lima, S. R. & Silva J. M. (2017) *Challenges and trends for sampling-based monitoring in SDN.* University of Minho.

Edwards, J. (2020, 30 juni). Smart network upgrades to consider before the next pandemic. *Networkworld*. https://www.networkworld.com/article/3564544/smart-network-upgrades-to-consider-before-the-next-pandemic.html

Eidenskog, D. & Karresand, M. (2017). *Risker med virtualisering av IT-system* (FOI-R--4448--SE). Totalförsvarets forskningsinstitut (FOI).

ETSI (2020), *Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV* (ETSI GR NFV 003 V1.5.1, 2020-01).

Extreme Networks (2019). *SD-LAN vs LAN: What Are The Key Differences?* (Blog post). https://www.extremenetworks.com/extreme-networks-blog/sd-lan-vs-lan-what-are-the-key-differences/

Fairfield, E. (2020, 23 september). Cisco ACI or VMware NSX – Why Not Both? ). https://www.wwt.com/article/cisco-aci-or-vmware-nsx-why-not-both

Feamster, N., Rexford, J. och Zegura, E. (2014). The road to SDN: an intellectual history of programmable networks. *ACM SIGCOMM Computer Communication Review*, 44(2), 87–98. https://doi.org/10.1145/2602204.2602219

Graham-Cumming, J. (2019). *Details of the Cloudflare outage on July 2, 2019*. The Cloudflare Blog. https://blog.cloudflare.com/details-of-the-cloudflare-outage-on-july-2-2019/

GSM Association (2017). *An Introduction to Network Slicing*. https://www.gsma.com/futurenetworks/wp-content/uploads/2017/11/GSMA-An-Introduction-to-Network-Slicing.pdf

Guo, L., Pang, J., & Walid, A. (2016, November). Dynamic service function chaining in SDN-enabled networks with middleboxes. In *2016 IEEE 24th International Conference on Network Protocols* (ICNP) (pp. 1–10). IEEE. https://doi.org/10.1109/ICNP.2016.7784431

Gustafsson, T. Almroth, J. & Mörnestedt, F. (2012). *Reaktiva nät*. FOI-R--3560--SE.

Haleplidis, E. (red), Pentikousis, K. (red), Denazis, S., Hadi Salim, J., Meyer, D. & Koufopavlou, O. (2015). *Software-Defined Networking (SDN): Layers and Architecture Terminology* (RFC 7426). Internet Engineering Task Force (IETF). https://tools.ietf.org/html/rfc7426

Hares, S., Lopez, D., Zarny, M., Jacquenet, D., Kumar, R., & Jeong, J. (2017). *Interface to Network Security Functions (I2NSF): Problem Statement and Use Cases* (RFC 8192). https://tools.ietf.org/html/rfc8192

Hedlund, B. (2013) *VMware NSX, Convergence, and Reforming Operational Visibility for the SDDC*. VMware. https://blogs.vmware.com/networkvirtualization/2013/10/vmware-nsx-visibility-sddc.html/

Hock, D., Hartmann, M., Gebert, S., Jarschel, M., Zinner, T., & Tran-Gia, P. (2013). Pareto-optimal resilient controller placement in SDN-based core networks. In *Proceedings of the 2013 25th International Teletraffic Congress* (ITC) (pp. 1–9). IEEE. https://doi.org/10.1109/ITC.2013.6662939

Hong, S., Xu, L., Wang, H., & Gu, G. (2015). Poisoning network visibility in software-defined networks: New attacks and countermeasures. In *2015 Network and Distributed System Security Symposium* (NDSS) (Vol. 15, pp. 8–11). https://doi.org/10.14722/NDSS.2015.23283

IEEE (1999). *IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks* (IEEE Std 802.1Q-1998). https://ieeexplore.ieee.org/document/753056

IEEE (2016). *IEEE Standard for Local and metropolitan area networks – Station and Media Access Control Connectivity Discovery* (IEEE Std 802.1AB-2016). https://ieeexplore.ieee.org/browse/standards/get-program/page/series?id=68

IEEE (2018). *IEEE Standard for Local and Metropolitan Area Networks – Bridges and Bridged Networks* (IEEE Std 802.1Q-2018). https://ieeexplore.ieee.org/browse/standards/get-program/page/series?id=68

International Organization for Standardization (1994). *Information technology — Open Systems Interconnection — Basic Reference Model: The Basic Model* (ISO/IEC-standard nr. 7498-1:1994). https://www.iso.org/standard/20269.html

Jain, V., Yatri, V., Kanchan, & Kapoor, C. (2019). Software defined networking: State-of-the-art. *Journal of High Speed Networks*, 25(1), 1–40. IOS press. https://doi.org/10.3233/JHS-190601

Jeong, J., Hyun, S., Ahn, T., Hares, S. och Lopez, D. (2019). *Applicability of Interfaces to Network Security Functions to Network-Based Security Services* (draft-ietf-i2nsf-applicability-18). Internet Engineering Task Force (IETF). https://datatracker.ietf.org/doc/draft-ietf-i2nsf-applicability/

John, W., Pentikousis, K., Agapiou, G., Jacob, E., Kind, M., Manzalini, A., Risso, F., Staessens, D. Steinert, R, & Meirosu, C. (2013). Research Directions in Network Service Chaining. In *2013 IEEE SDN for Future Networks and Services* (SDN4FNS). https://doi.org/10.1109/SDN4FNS.2013.6702549

Juniper (2019). *Contrail Networking Architecture Guide Detailed Technical Description of the Contrail Virtual Networking and Security Platform*. https://www.juniper.net/documentation/en_US/release-independent/solutions/information-products/pathway-pages/sg-010-contrail-networking-arch-guide.pdf

Kamath, S., Singh, S., & Kumar, M. S. (2019). Multiclass queueing network modeling and traffic flow analysis for SDN-enabled mobile core networks with network slicing. *IEEE Access*, 8, 417–430. https://doi.org/10.1109/ACCESS.2019.2959351

Kerraval, Z. (2020, 5 maj). Nvidia’s aggressive purchases could signal the era of open networking [Opinion]. *Network World*. https://www.networkworld.com/article/3541824/nvidias-aggressive-purchases-could-signal-the-era-of-open-networking.html

Kousalya, G., Balakrishnan, P., & Raj, C. P. (2017). Demystifying the Traits of Software-Defined Cloud Environments (SDCEs). In *Automated Workflow*

*Scheduling in Self-Adaptive Clouds Computer* (pp. 23–53). Springer. https://doi.org/10.1007/978-3-319-56982-6_2

Kreutz, D., Ramos, F. M. V., Verissimo, P., Rothenberg, C. E., Azodolmolky, S. och Uhlig, S. (2015). Software-Defined Networking: A Comprehensive Study. *Proceedings of the IEEE*, 103(1), 14–76. IEEE. https://doi.org/10.1109/JPROC.2014.2371999

Kuzniar, M., Peresini, P., Canini, M., Venzano, D., & Kostic, D. (2012). A SOFT way for openflow switch interoperability testing. In *Proceedings of the 8th international conference on Emerging networking experiments and technologies* (pp. 265–276). https://doi.org/10.1145/2413176.2413207

Latif, Z., Sharif, K., Li, F., Karim, M. M., Biswas, S., & Wang, Y. (2020). A comprehensive survey of interface protocols for software defined networks. *Journal of Network and Computer Applications*, 156. https://doi.org/10.1016/j.jnca.2020.102563

Lerner, A. (2017, 7 februari). *Intent-based networking*. Gartner blog. https://blogs.gartner.com/andrew-lerner/2017/02/07/intent-based-networking/

Li, Z. (2018). The Main Contents and the Architecture of the Telecommunication 4.0. In *Telecommunication 4.0* (pp. 27-72). Springer. https://doi.org/10.1007/978-981-10-6301-5_3

Li, Y., & Chen, M. (2015). Software-defined network function virtualization: A survey. *IEEE Access*, 3, 2542-2553. https://doi.org/10.1109/ACCESS.2015.2499271

Lopez, D., Lopez, E., Dunbar, L., Strassner, J. och Kumar, R. (2018). *Framework for Interface to Network Security Functions* (RFC 8329). Internet Engineering Task Force (IETF). https://tools.ietf.org/html/rfc8329

Mishra, V., Verma, D., & Williams, C. (2016). *Leveraging sdn for cyber situational awareness in coalition tactical networks* (STO-MP-IST-148). Nato. https://www.sto.nato.int/publications/STO%20Meeting%20Proceedings/STO-MP-IST-148/MP-IST-148-02.pdf

Mordor Intelligence (u.å.). Network slicing market - growth, trends, and forecast (2020 - 2025). https://www.mordorintelligence.com/industry-reports/network-slicing-market

Musa, S. (2018). *Network Security and Cryptography*. Mercury Learning and Information.

Mämmelä, O., Hiltunen, J., Suomalainen, J., Ahola, K., Mannersalo, P. & Vehkaperä, J. (2016) *Towards Micro-Segmentation in 5G Network Security*. EuCNC 2016.

Odom, W. (2020). Introduction to Controller-Based Networking (fritt tillgängligt kapitel ur *CCNA 200-301 Official Cert Guide, Volume 2*). Cisco Press. https://www.ciscopress.com/articles/article.asp?p=2995354

Oltsik, J. (2017) Micro-segmentation projects span enterprise organizations. Network World (Online); Southborough. Network World Inc.

Orange (2019, 4 februari). SDx: software-defined everything comes to the LAN. *Orange Business Services Magazine*. https://www.orange-business.com/en/magazine/sdx-software-defined-everything-comes-lan

Pakzad, F., Portmann, M., Tum, W. L. & Indulska, J. (2016). Efficient topology discovery in OpenFlow-based Software Defined Networks. *Computer Communications*, 77, s. 52–61. https://doi.org/10.1016/j.comcom.2015.09.013

Palo Alto Networks (u.å.). *What is microsegmentation?* https://www.paloaltonetworks.com/cyberpedia/what-is-microsegmentation

Perrow, C. (1984). *Normal accidents: Living with high risk technologies*. BasicBooks.

Petroulakis, N. E., Fysarakis, K., Askoxylakis, I., & Spanoudakis, G. (2018). Reactive security for SDN/NFV ‐ enabled industrial networks leveraging service function chaining. *Transactions on Emerging Telecommunications Technologies*, 29(7). https://doi.org/10.1002/ett.3269

Postel, J. & Reynolds, J. (1988). *A Standard for the Transmission of IP Datagrams over IEEE 802 Networks* (RFC 1042). Internet Engineering Task Force (IETF). https://tools.ietf.org/html/rfc1042

Pueblas, M., Gyurindak, S. Strika, J., Kachalia, R., Hamilton, D., & Tenneti, S. (2010). *Small Enterprise Design Profile Reference Guide*. Cisco.

Queiroz, W., Capretz, M. A., & Dantas, M. (2019). An approach for SDN traffic monitoring based on big data techniques. *Journal of Network and Computer Applications*, 131, 28–39. https://doi.org/10.1016/j.jnca.2019.01.016

Raj, P., & Raman, A. (2018). *Software-defined Cloud Centers*. Springer. https://doi.org/10.1007/978-3-319-78637-7

Rapp, J. (2019) Evolution of Software-Defined Networking for dummies. Vmware. Research Desk.

Reith, L. (2019). Paradigm Shift! The Path to Brutal Automation. In *Future Telco* (pp. 111–129). Springer. https://doi.org/10.1007/978-3-319-77724-5_10

Saint-Andre, P. (2011). *Extensible Messaging and Presence Protocol (XMPP): Core* (RFC 6120). Internet Engineering Task Force (IETF). https://tools.ietf.org/html/rfc6120

Sandhya, Sinha, Y. och Haribabu, K. (2017). A survey: Hybrid SDN. *Journal of Network and Computer Applications*, 100, s. 35–55. http://dx.doi.org/10.1016/j.jnca.2017.10.003

Scott-Hayward, S., O'Callaghan, G., & Sezer, S. (2013). SDN security: A survey. In *2013 IEEE SDN For Future Networks and Services* (SDN4FNS). IEEE. https://doi.org/10.1109/SDN4FNS.2013.6702553

sFlow (n.d) *About sFlow*. https://sflow.org/about/index.php

Shameli-Sendi, A. S., Jarraya, Y., Pourzandi, M., & Cheriet, M. (2016). Efficient provisioning of security service function chaining using network security defense patterns. *IEEE Transactions on Services Computing*, 12(4), 534–549. IEEE. https://doi.org/10.1109/TSC.2016.2616867

Shin, S., Xu, L., Hong, S., & Gu, G. (2016). Enhancing network security through software defined networking (SDN). In *2016 25th international conference on computer communication and networks* (ICCCN). IEEE. https://doi.org/10.1109/ICCCN.2016.7568520

Shirali-Shahreza, S. & Ganjali, Y. (2013). Efficient Implementation of Security Applications in OpenFlow Controller with FleXam. In *2013 IEEE 21st Annual Symposium on High-Performance Interconnects*. IEEE. https://doi.org/10.1109/HOTI.2013.17

Singhal, A. & Jain, R. (2002). Terabit switching: a survey of techniques and current products. *Computer Communications*, 25, 547–556. https://doi.org/10.1016/S0140-3664(01)00423-6

Šulák, V., Helebrandt, P., & Kotuliak, I. (2016). Performance analysis of OpenFlow forwarders based on routing granularity in OpenFlow 1.0 and 1.3. In *2016 19th Conference of Open Innovations Association* (FRUCT) (pp. 236–241). IEEE. https://doi.org/10.23919/FRUCT.2016.7892206

T-Systems (2018). *SD-LAN: Evolving technology that will change the way customers monetize their enterprise networks* (white paper). https://www.t-systems.com/blob/890854/7480dddd348f0a7286b0ffa419aaca56/DL_WP_SD-LAN.pdf

Venugopal, V., Alves-Foss, J., & Ravindrababu, S. G. (2019). Use of an SDN Switch in Support of NIST ICS Security Recommendations and Least Privilege Networking. In *Proceedings of the Fifth Annual Industrial Control System Security (ICSS) Workshop* (pp. 11–20). ACM. https://doi.org/10.1145/3372318.3372321

Vmware (2014) Data Center Micro-Segmentation: A Software Defined Data Center Approach for a ”Zero Trust” Security Strategy. White paper. Vmware.

Vmware (2019) *IPFIX*. https://docs.vmware.com/en/VMware-vRealize-Network-Insight/5.1/com.vrni.faq.doc/GUID-A5AEB90E-A3C2-45D7-BD5A-B23E45F98687.html

Wanga, P., Yanga, L. T. Niea X., Rena Z., Lia J. & Kuang L. (2020) Data-driven software defined network attack detection: State-of-the-art and perspectives. *Information Sciences*, 513, 65–83. https://doi.org/10.1016/j.ins.2019.08.047

Wood, M. (2017a). Top requirements on the SD-WAN security checklist. *Network Security*, 2017(7), 9–11. https://doi.org/10.1016/S1353-4858(17)30070-3

Wood, M. (2017b). How to make SD-WAN secure. *Network Security*, 2017(1), 12–14. https://doi.org/10.1016/S1353-4858(17)30006-5

Zhang, N., Yang, P., Zhang, S., Chen, D., Zhuang, W., Liang, B., & Shen, X. S. (2017). Software defined networking enabled wireless network virtualization: Challenges and solutions. *IEEE Network*, 31(5), 42–49. https://doi.org/10.1109/MNET.2017.1600248

Zhu, S.Y., Scott-Hayward, S., Jacquin, L. och Hill, R. (2017) *Guide to Security in SDN and NFV - Challenges, Opportunities, and Applications*. Springer. https://doi.org/10.1007/978-3-319-64653-4

Mjukvarudefinierade nätverk ändrar modellen för hur trafikflöden styrs i nätverk genom att styrlogiken centraliseras och samtidigt separeras från nätverkskomponenternas datavägar. Tekniken syftar till att möjliggöra administrativt enklare styrning av trafikflöden, som dessutom kan ske på högre abstraktionsnivå. Mjukvarudefinierade nätverk är väl etablerade på marknaden, vilket innebär att flera populära kommersiella lösningar använder tekniken. Mjukvarudefinierade nätverk är vanliga i datacenter och trenden tycks vara att tekniken blir allt vanligare även i enklare systemlösningar.

Syftet med denna studie är att ge läsaren en grundläggande kunskap om mjukvarudefinierade nätverk och de byggstenar som används i dem. Kunskapen är främst avsedd att underlätta diskussioner kring mjukvarudefinierade nätverk i samband med utveckling och förvaltning av IT-system inom Försvarsmakten och andra offentliga organisationer.

Rapporten tar upp vad mjukvarudefinierade nätverk är, vilka byggstenar som krävs för att bygga sådana nätverk, hur de kan påverka nätverken ur olika aspekter och vilka effekter de kan få i IT-systemperspektivet. Rapporten har sin grund i vetenskaplig litteratur, som till största del centrerar kring öppna lösningar för mjukvarudefinierade nätverk, då information saknas i djupet om proprietära lösningar.

Många av de problem som hanteras med hjälp av mjukvarudefinierade nätverk kan även lösas med andra tekniker. Mjukvarudefinierade nätverk för dock även med sig andra fördelar, exempelvis underlättandet av administrationen av nätverket. Dessutom förbättrar tekniken möjligheterna för nyttjandet av andra lösningar så som tjänstekedjor eller mikrosegmentering. Mjukvarudefinierade nätverk för dock inte endast med sig fördelar utan introducerar även nya angreppsytor och fallgropar, såsom att den centrala styrlogiken blir ett attraktivt mål för angripare.

FOI

ISSN 1650-1942

www.foi.se