فصلنامه علمی - پژوهشی فرماندهی و کنترل، سال پنجم، شماره سه، پاییز ۱۴۰۰

بررسی تاثیر شبکه نرم افزار محور بر مولفه‌های کیفیت سرویس در شبکه

محمدرضا مصلحی[۱]، حسین ابراهیم پور کومله[۲]، سلمان گلی بید گلی[۳]

تاریخ دریافت: ۱۴۰۰/۰۷/۱۵ **تاریخ پذیرش: ۱۴۰۰/۰۹/۲۹**

چکیده

یکی از مشکلات شبکه‌های رو به رشد امروزی نظیر شبکه‌های اینترنت اشیاء، نیاز به برآوردن سطوح مشخصی از مولفه‌های کیفیت سرویس است. رشد بسیار سریع تعداد دستگاه‌های متصل در شبکه‌های امروزی نظیر اینترنت اشیاء، شامل اشیاء هوشمندی مثل گوشی‌های هوشمند، حسگرها، عملگرها و دیگر تجهیزات، سبب ایجاد و توسعه خدمات و کاربردهای جدیدی از جمله در حوزه‌های شهرها و خانه‌های هوشمند شده است. این خدمات و کاربردهای نوظهور و رو به گسترش، هر کدام نیازهای کیفیت سرویس مختص به خود را دارند و برآورده کردن نیازهای کیفیت سرویس آنها با استفاده از زیرساخت‌های مرسوم شبکه، که تنها مبتنی بر عملکرد سخت افزاری هستند امکان پذیر نبوده و یا به سختی قابل دست‌یابی است. یکی از راه حل‌های این مشکل ایجاد زیر ساخت شبکه بر اساس شبکه‌های نرم افزار محور به عنوان یک راه حل امید بخش می‌باشد. این فناوری با جدا کردن صفحه کنترل از صفحه ارسال قابلیت نظارت خودکار و مدیریت مبتنی بر سیاست‌گذاری در زیرساخت و منابع شبکه را فراهم می‌کند. بر این اساس در این مقاله یک زیر ساخت مبتنی بر شبکه نرم افزار محور را پیاده سازی کرده و پس از ارزیابی، نتایج اندازه گیری پارامترهای کیفیت سرویس را با نتایج آزمایش مشابه در زیر ساخت مرسوم شبکه مقایسه می‌کنیم. نتایج شبیه سازی نشان دهنده برتری زیرساخت شبکه نرم افزار محور می‌باشد.

واژگان کلیدی: شبکه نرم افزار محور، کیفیت سرویس، اینترنت اشیاء،

[۱] عضو هیات علمی (موسسه آموزش عالی جهاد دانشگاهی، مربی) moslehi@acecr.ac.ir (نویسنده مسئول)

[۲] عضو هیات علمی (دانشگاه کاشان، استادیار) ebrahimpoor@kashanu.ac.ir

[۳] عضو هیات علمی (دانشگاه کاشان، استادیار) ebrahimpoor@kashanu.ac.ir

۱. مقدمه

با رشد و توسعه سریع فناوری‌های اینترنت و شبکه‌های کامپیوتری از جمله اینترنت اشیاء[۴]، شاهد ظهور مفاهیم جدیدی در کاربردهای شبکه هستیم. پیامد این رشد، افزایش تعداد دستگاه‌های متصل مانند گوشی‌های هوشمند و افزایش کاربران است. بر اساس گزارش سالانه سیسکو [۱] تا سال ۲۰۲۳، تعداد دستگاه‌های متصل به تعداد ۱٤,۷ میلیارد دستگاه خواهد رسید. بر اساس همین گزارش تعداد کل مشترکان تلفن همراه جهانی به ۵,۷ میلیارد رسیده و بیش از ۷۰ درصد جمعیت جهان از اتصال تلفن همراه برخوردار خواهند بود. این افزایش چالش-های بزرگی را در شبکه‌های مربوطه ایجاد می‌کند. مهم‌ترین چالش این رشد سریع، پیچیدگی مدیریت شبکه و عدم مقیاس پذیری آنها است. مشکلات دیگر می‌توان به افزایش، تاخیر[۵] و ازدحام[۶] و ترافیک شبکه اشاره کرد. جهت ارایه خدمات در شبکه‌های امروزی، درخواست منابع از طرف برنامه‌های کاربردی، از دستگاه‌ها و تجهیزات کاربر نهایی در مقیاس زیاد و از مکان‌های مختلف ارسال می‌شوند. این برنامه‌ها و خدمات عموما نیازمندی‌های کیفیت سرویس[۷] سخت‌گیرانه‌ای مانند نیاز به تأخیر کم را دارا هستند. در شبکه های امروزی، برخی از برنامه‌ها و خدمات در حوزه اینترنت اشیاء و شهر هوشمند مانند سلامت الکترونیک و حتی امنیت و انرژی نه تنها مفید هستند، بلکه برای ارائه خدمات در زمان مورد انتظار با پارامترهای کیفیت سرویس بالاتر، حیاتی هستند. با فناوری‌های مرسوم شبکه، این چالش‌ها و نیازمندی‌ها نمی‌توانند برآورده شوند.

برای برآورده کردن این نیازمندی‌های سخت‌گیرانه، لازم است که فناوری‌های شبکه با استفاده از مفاهیم جدیدی مانند شبکه‌های نرم‌افزار محور[۸] و دیگر فناوری‌هایی مثل محاسبات لبه چند دسترسی[۹] با نیازهای آینده سازگار شوند. بنیاد شبکه های باز[۱۰] شبکه نرم افزار محور را به عنوان یک معماری نوظهور و مناسب برای کاربردهای امروزی، با نیاز به پهنای باند بالا و ماهیت پویا تعریف می‌کند. ویژگی‌های پویایی، قابلیت مدیریت، مقرون به صرفه بودن و قابلیت سازگاری این فناوری مهم‌ترین عوامل تاثیر گذاری آن برای نیازهای امروزی است [۲]. معماری شبکه نرم افزار محور، صفحه کنترل شبکه را از صفحه ارسال یا داده جدا می‌کند. با جداسازی صفحه کنترل شبکه و صفحه داده یا ارسال، می‌توانیم عملکرد کنترل شبکه را بطور مستقیم برنامه‌ریزی کنیم. شبکه نرم افزار محور می‌تواند به طور متمرکز کل شبکه را با برنامه نویسی پویا، کنترل و مدیریت کند.

یک کنترل کننده متمرکز به عنوان قلب شبکه می‌تواند عملکرد شبکه را به صورت پویا بهینه کند. در کنار فناوری شبکه نرم افزار محور، و مطابق با تعریف ETSI[۱۱]، محاسبات لبه چند دسترسی، به توسعه دهندگان برنامه و ارائه دهندگان محتوا قابلیت‌های محاسبات و خدمات در لبه شبکه را ارائه می‌دهد [۳]. مشخصات لبه شبکه، تأخیر بسیار کم و پهنای باند بالا و همچنین دسترسی بلادرنگ عمدتا بصورت بی سیم است که می‌تواند توسط برنامه‌هایی با الزامات بلادرنگ و تاخیر کم مورد استفاده قرار گیرد [٤]. ETSI با همکاری کنسرسیوم OpenFog در حال حاضر استانداردهایی را برای API در بخش لبه شبکه وضع کرده است. ETSI یک گروه مشخصات صنعتی[۱۲] با وظیفه ایجاد استانداردهای محاسبات لبه را تشکیل داده تا مزایای آن را برای همه اجزای شبکه بهینه کند. هدف اصلی بر همکاری بین اپراتورهای شبکه، برنامه‌های کاربردی و ارائه دهندگان محتوا برای تقویت تجربه کلی کاربر متمرکز است. محیط‌های اینترنت اشیا مانند شهرهای هوشمند شامل تعداد زیادی حسگر ناهمگن، محرک‌ها، گوشی‌های هوشمند و دیگر تجهیزاتی است که به زیرساخت‌های شبکه متصل می‌شوند، بنابراین برای مدیریت و کنترل پویای دستگاه‌ها و شبکه‌ها به یک معماری انعطاف‌پذیر نظیر فناوری شبکه نرم افزار محور نیاز دارد. در

[۴] Internet of Things
[۵] Delay
[۶] Congestion
[۷] Quality of Service
[۸] Software-Defined Networking - SDN
[۹] Multi-access Edge Computing
[۱۰] Open Networking Foundation
[۱۱] European Telecommunications Standards Institute
[۱۲] Industry Specification Group

این مقاله ابتدا یک مدل با استفاده از شبکه نرم افزار محور ارایه و سپس عملکرد این فناوری را با فناوری مرسوم شبکه ارزیابی و مقایسه می‌کنیم. در واقع هدف اصلی ارزیابی عملکرد شبکه تحت انواع مختلف ترافیک و مطالعه نقش یک کنترل کننده متمرکز است. بقیه مقاله به شرح زیر سازماندهی شده است: بخش دوم برخی از آثار مرتبط را بررسی می‌کند. در بخش سوم مروری کوتاه بر فناوری شبکه نرم افزار محور ارایه داده شده است. بخش چهارم مدل شبکه مبتنی بر کنترل کننده شبکه نرم افزار محور را پیشنهاد داده شده است. بخش پنجم نتایج را تجزیه و تحلیل می‌کند و نتیجه گیری مقاله در بخش ششم ارایه شده است.

۲. کارهای مرتبط

سعید و همکاران [۵] با استفاده از زیر ساخت شبکه نرم افزار محور و بکار گیری مفهوم محاسبات مه[۱۳] یک زمان بند وظایف امن پیشنهاد داده‌اند. الگوریتم آنها به نام FUPE در زیرساخت شبکه نرم افزار محور با استفاده از رویکرد فازی با هدف تامین سطح مناسبی از امنیت پیاده سازی شده است. آنیکور و همکاران [٦] با توجه به پاندمی کووید ۱۹ و لزوم فاصله گذاری اجتماعی یک چارچوب خودکار و هوشمند برای معماری صنعتی Industry 4.0 در اینترنت اشیاء و با استفاده از شبکه‌های نرم افزار محور و به منظور انجام عملیات مورد نیاز از راه دور پیشنهاد داده‌اند. محمد و همکاران [۷] با استفاده از زیرساخت ایجاد شده توسط شبکه نرم افزار محور، یک الگوریتم یادگیری تقویتی عمیق برای تحویل محتوا در شبکه اینترنت اشیاء پیشنهاد داده و آن کارایی آن را در مقایسه با دیگر روش های مشابه نشان داده‌اند. تریفون و همکاران [۸] یک راه حل شبکه نرم افزار محور برای شبکه حسگر بی سیم به نام CORAL-SDN معرفی می‌کند. در روش پیشنهادی آنها سعی شده مشکلاتی را که با یکپارچه‌سازی شبکه نرم افزار محور با شبکه حسگر بی سیم به وجود می‌آید حل کند. یکی از این مشکلات، افزایش مقدار بسته‌های کنترلی تولید شده توسط شبکه نرم افزار محور است که می‌تواند در کیفیت ارتباطات رادیویی تاثیر گذار باشد. دوسینه و همکاران [۹] یک معماری SDN/NFV[۱۴] برای برآورده کردن الزامات اینترنت اشیاء برای استقرار چارچوب اینترنت اشیاء پیشنهاد می‌کند. معماری پیشنهادی آنها قادر است کل شبکه را توسط برنامه های کنترل کننده شبکه نرم افزار محور هماهنگ کند. آنها نقش SDN/NFV را در استقرار خدمات اینترنت اشیاء مطالعه کردند. اینتیدار و همکاران [۱۰] عملکرد معماری شبکه مبتنی بر شبکه نرم افزار محور را با معماری سنتی شبکه تجزیه و تحلیل و مقایسه کرده‌اند. در این کار آنها از کنترلرهای مختلف که در سه توپولوژی به کار گرفته شده، و ترافیک را از پروتکل-های مختلف ارسال می‌کنند، بهره برده‌اند. پس از اندازه گیری برخی از معیارهای شبکه، آنها عملکرد شبکه‌های زیربنایی را با هم مقایسه کرده‌اند. رائول و همکاران [۱۱] یک معماری شبکه نرم افزار محور چند لایه و معماری هماهنگ ساز لبه/ابر[۱۵] را پیشنهاد می‌کنند. معماری آنها یک روش کنترل ترافیک و جلوگیری از ازدحام برای اینترنت اشیاء، با در نظر گرفتن توزیع پویای پردازش از دید منابع واقعی شبکه را به کار می‌گیرد. چوان فنگ زو و همکاران [۱۲] با توجه به طبقه بندی ترافیک (DDTC) برای شهرهای هوشمند، یک استراتژی دفاعی حمله DDoS پیشنهاد کرده است. آنها برای افزایش انعطاف‌پذیری و کاهش بار شبکه نرم افزار محور در برابر حملات DDoS، از مجازی‌سازی عملکرد شبکه نرم افزار محور یا SDN/NFV با مکانیزمی برای طبقه‌بندی ترافیک استفاده می‌کنند. چوان لین و همکاران [۱۳] برای برنامه ریزی انتقال ترافیک حساس به تاخیر، یک موتور مهندسی ترافیک به نام DTE-SDN با استفاده از شبکه نرم افزار محور پیشنهاد کرده‌اند. DTE-SDN معیارهای کیفیت سرویس مانند توان عملیاتی و تاخیر هر اتصال شبکه را با گرفتن نمای کلی از شبکه بصورت بلادرنگ که با استفاده از پروتکل OpenFlow امکان پذیر است، نظارت می‌کند. شن وانگ و همکاران [١٤] با استفاده از یادگیری انتقالی و استانداردهای IEC 61850، مکانیزمی به نام SSDS یا امنیت نرم‌افزار محور هوشمند برای یکی از مهم‌ترین مؤلفه‌های شهرهای هوشمند به

[۱۳] Fog Computing

[۱۴] Software-Defined Networking/Network Function Virtualization

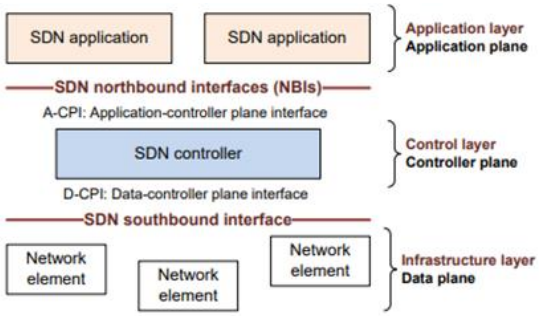
[۱۵] Edge/Cloud

نام وسیله نقلیه به شبکه یا V2G[۱۶] را با هدف ذخیره‌سازی انرژی و برنامه‌ریزی را پیشنهاد می‌کنند.

۳. مروری بر شبکه نرم افزار محور

این بخش یک نمای کلی از معماری شبکه نرم افزار محور ارائه می‌دهد. شبکه‌های مرسوم و سنتی سخت افزار محور هستند. در یک شبکه سنتی بزرگ، پیکربندی مجدد دستگاه‌ها و تجهیزات شبکه مانند مسیریاب‌ها و سوئیچ‌ها برای مدیران شبکه بسیار سخت است و از این رو مدیریت و کنترل شبکه و نظارت بر منابع شبکه[۱۷]و مولفه‌های کیفیت سرویس، کارهای بسیار پیچیده‌ای هستند. در شبکه‌های اینترنت اشیاء و شهرهای هوشمند با ترافیک حساس به زمان و ناهمگن مواجه هستیم و کاهش مصرف پهنای باند[۱۸] و سایر نیازهای مرتبط با زمان‌بندی مانند توان عملیاتی[۱۹]، از دست دادن بسته‌ها[۲۰]، تأخیر و جیتر[۲۱] در آن شبکه‌ها مهم است.

معماری شبکه نرم افزار محور به عنوان یک تحول در فناوری شبکه پدید آمده است و هدف آن ایجاد شبکه‌هایی با انعطاف پذیری بیشتر و مدیریت بهتر با پیچیدگی کمتر است. ایده اصلی شبکه نرم افزار محور جدا کردن صفحه کنترل[۲۲] از صفحه داده[۲۳] است. این فناوری برنامه‌ریزی، چابکی و انعطاف‌پذیری بهتر را در شبکه امکان پذیر کرده و قابلیت مدیریت متمرکز و یا توزیع شده شبکه را فراهم می‌کند [۲]. با یک کنترل کننده متمرکز در شبکه نرم افزار محور، مدیران شبکه می‌توانند به سرعت تغییرات در شبکه را مدیریت کنند.



شکل ۱ – مولفه های اصلی SDN [۲]

شبکه نرم افزار محور بر اساس تعریف بنیاد شبکه‌های باز دارای سه صفحه یا لایه است.

۱- صفحه یا لایه کاربرد؛ این صفحه در بالای معماری شبکه نرم افزار محور قرار دارد که از یک یا چند برنامه کاربردی سمت کاربر نهایی (مهندسی ترافیک[۲۴]، امنیت و غیره [۱۵]) تشکیل شده است که با کنترل کننده (ها) برای استفاده از نمای انتزاعی شبکه برای فرآیند تصمیم گیری داخلی آنها تعامل دارند [۱۶]. وظیفه اصلی، ارایه تعریفی از الزامات و رفتارهای شبکه است.

۲- صفحه یا لایه کنترل کننده؛ صفحه کنترل بین برنامه-های کاربردی و صفحه داده قرار گرفته است و به عنوان یک لایه واسط بین برنامه‌های کاربردی از طریق رابط شمالی[۲۵] و صفحه داده از طریق رابط جنوبی[۲۶] عمل می‌کند [۱۷]. وظیفه این صفحه، تفسیر الزامات از صفحه کاربردی به صفحه داده است. این صفحه شامل کنترل کننده‌های شبکه نرم افزار محور است که دستگاه‌های صفحه داده را کنترل می‌کنند. یک کنترل کننده دارای دو جزء اصلی یکی به نام مولفه

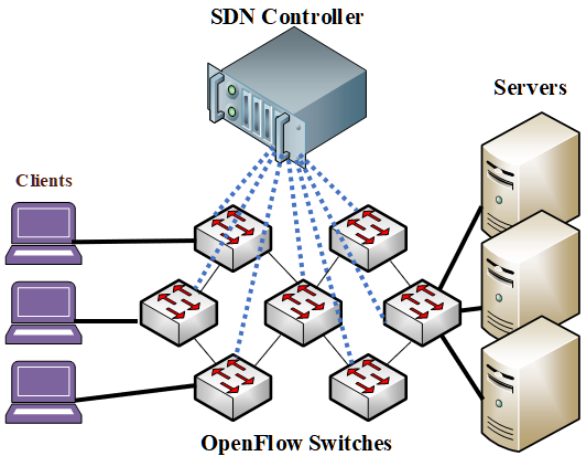
[۱۶] Vehicle-to-Grid
[۱۷] Network Resources
[۱۸] Bandwidth
[۱۹] Throughput
[۲۰] Packet Loss
[۲۱] Jitter
[۲۲] Control Plane
[۲۳] Data Plane
[۲۴] Traffic Engineering
[۲۵] North-bound Interface
[۲۶] South-bound Interface

عملکردی[۲۷] و دیگری به نام منطق کنترل[۲۸] است. مولفه عملکردی مسئول مدیریت رفتار کنترل کننده است و منطق کنترل مسئول مسائل مربوط به دستورالعمل‌های منابع شبکه برای تأمین نیازهای برنامه‌های کاربردی در شبکه نرم افزار محور است [۲].

۳- صفحه یا لایه داده (لایه زیرساخت)؛ صفحه داده پایین‌ترین صفحه معماری شبکه نرم افزار محور است. این صفحه شامل دستگاه‌های فیزیکی و مجازی (مانند مسیریاب‌ها، سوئیچ‌ها و نقاط دسترسی) است که وظایف ارسال و مسیریابی را انجام می‌دهند. دستگاه‌های شبکه را می‌توان از طریق یک اتصال امن به کنترل کننده مرکزی شبکه نرم افزار محور متصل کرد. کنترل کننده، قوانین را مشخص می‌کند و آنها را برای ایجاد جداول ارسال به دستگاه‌های شبکه ارسال می‌کند. یکی از محبوب‌ترین پروتکل‌های کنترلی که برای اتصال دستگاه‌ها و کنترل کننده‌های صفحه داده استفاده می‌شود پروتکل OpenFlow [۱۸] است.

٤. معماری مبتنی بر شبکه نرم افزار

در شبکه‌های امروزی تعداد زیادی دستگاه شامل کامپیوترها، حسگرها، محرک‌ها، گوشی‌های هوشمند و غیره، می‌توانند به تبادل داده بپردازند. برنامه‌های بسیار زیادی در کاربردهای امروزی با نیازهای کیفیت سرویس مانند سطوح مختلف از پهنای باند، توان عملیاتی و تاخیر و نظائر آن وجود دارد. از سوی دیگر، این تعداد بسیار زیاد دستگاه، حجم عظیمی از داده را تولید خواهند کرد. حجم انبوه داده چالش بزرگی است که بر روی ارضای نیازهای کیفیت سرویس تأثیر می‌گذارد. با توجه به ویژگی ایستایی فناوری‌های مرسوم شبکه از یک طرف و پویایی محیط‌های شبکه‌های امروزی از طرف دیگر، مدیریت و تخصیص منابع شبکه با لحاظ نمودن کیفیت سرویس با آن فناوری ها یک کار چالش برانگیز است. سوئیچ‌های لایه دو و لایه سه معمولی کار یکسانی را برای هر بسته ورودی انجام می‌دهند. در آن دستگاه‌ها عملیات ارسال برای هر بسته به صورت جداگانه و مبتنی بر بسته انجام می‌شود. در شبکه‌های مبتنی بر شبکه نرم افزار محور، عملیات ارسال مبتنی بر جریان-های ترافیکی انجام می‌شود. جریان‌ها مجموعه‌ای از دنباله‌های بسته‌ها برای اتصالات انتها به انتهای مشخصی هستند. با فناوری شبکه نرم افزار محور و پروتکل OpenFlow، می‌توان رفتار ارسال را برای هر یک از آن اتصالات انتها به انتها در سوئیچ ها تعریف کرد و از این رو عملیات سوئیچ ها در شبکه نرم افزار محور، ارسال مبتنی بر جریان نامیده می‌شود. اخیراً، در پژوهش-های متعددی برای غلبه بر چالش‌های شبکه‌ها، از جمله کیفیت سرویس از تکنیک‌های مبتنی بر شبکه نرم افزار محور استفاده شده است. در این بخش، ما یک معماری مبتنی بر شبکه نرم افزار محور را مورد ارزیابی قرار داده تا عملکرد بهتر شبکه از نظر پهنای باند و تأخیر را به عنوان عوامل مهم معیارهای کیفیت سرویس بدست آوریم. با کنترل‌کننده شبکه نرم افزار محور متمرکز، ما یک سناریوی بسیار رایج و واقعی را که شامل اتصال بین دستگاه‌های نهایی بعنوان کلاینت به برخی منابع شبکه بعنوان سرور با کنترل‌کننده شبکه نرم افزار محور متمرکز است (شکل ۲) فرض می‌کنیم. با این سناریو، ما به دنبال ارزیابی فناوری شبکه نرم افزار محور در مقایسه با فناوری های مرسوم شبکه هستیم تا مزایای شبکه نرم افزار محور را برجسته کنیم.



شکل ۲. کنترل کننده متمرکز برای اتصال کلاینت–سرور

برای پوشش بیشتر پروتکل‌های مورد استفاده، ترافیک TCP و UDP را به عنوان ترافیک عمومی برای بسیاری از برنامه‌ها تولید

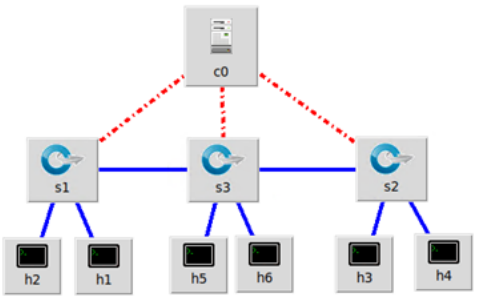
[۲۷] Functional component

[۲۸] Control Logic

می‌کنیم. برای کنترل‌کننده مرکزی، از یک کنترل‌کننده مبتنی بر پایتون به نام Ryu [۱۹] استفاده کرده‌ایم. Ryu یک سیستم عامل شبکه منبع باز و یک چارچوب سبک وزن و انعطاف پذیر برای توسعه برنامه‌های کاربردی شبکه نرم افزار محور است. این کنترل کننده مؤلفه‌های نرم‌افزار را با APIهای خوش تعریف ارائه می‌کند که توسعه دهندگان را قادر می‌سازد تا به راحتی برنامه‌های جدید مدیریت و کنترل شبکه را ایجاد نمایند. کنترلر Ryu از چندین پروتکل مانند OpenFlow، OF-config و Netconf برای مدیریت دستگاه‌های شبکه پشتیبانی می‌کند. Ryu به طور کامل از OpenFlow از جمله OpenFlow نسخه های ۱,۰ تا ۱,۵ پشتیبانی می‌کند. همه کدهای Ryu در پایتون تحت مجوز Apache 2.0 در دسترس هستند [۲۰]. در این مقاله از پروتکل OpenFlow نسخه ۱,۳ برای تعامل بین کنترلر و سوئیچ‌ها به عنوان دستگاه‌های صفحه ارسال استفاده شده است. در ادامه نتایج به دست آمده از ارزیابی عملکرد سناریوی مبتنی بر شبکه نرم افزار محور با نتایج ارزیابی عملکرد سناریوی مرسوم شبکه مقایسه خواهد شد.

۵. نتایج عملی

برای محیط شبیه‌سازی، از Mininet [۲۱] استفاده می‌کنیم، زیرا این شبیه ساز به طور گسترده برای محیط‌های آزمایش شبکه نرم افزار محور استفاده می‌شود. Mininet به عنوان یک شبیه ساز شبکه می‌تواند توپولوژی‌های مختلفی را با میزبان‌ها، سوئیچ‌های با قابلیت Openflow (یا OVSwitches) و لینک-های آنها ایجاد کند. شکل ۳ توپولوژی سناریوی مبتنی بر شبکه نرم افزار محور را نشان می‌دهد.



شکل ۳. توپولوژی مبتنی بر شبکه نرم افزار محور

ماشین‌های مجازی که برای اجرای سناریوها استفاده شده‌اند، روی رایانه‌ای که از پردازنده مرکزی Intel Core i7-4510 با فرکانس ۲,۶ گیگاهرتز و ۱۲ گیگابایت رم استفاده می‌کند، نصب شده‌اند. سناریوی شبکه مبتنی بر شبکه نرم افزار محور شامل سه سوئیچ با قابلیت Openflow (Open VSwitche یا OVS در mininet) و دو میزبان (کلاینت و سرور) متصل به هر یک از آنها که توسط یک کنترل‌کننده کنترل می‌شوند (کد پایتون شکل ۴) ایجاد شده است. ما از یک کنترل کننده Ryu بعنوان کنترل کننده مرکزی برای اجرای سناریوی مبتنی بر شبکه نرم افزار محور استفاده کرده-ایم. به منظور پیاده سازی کنترل کننده Ryu از "ryu.base.app_manager.RyuApp" استفاده شده و همچنین برای استفاده از نسخه OpenFlow 1.3، از روال "simple_switch_13.py"اعمال شده است. سناریوی مرسوم شبکه از شش میزبان و سه سوئیچ معمولی (بدون هیچ کنترل کننده ایی) تشکیل شده است. هر دو سناریو روی ماشین مجازی پیاده سازی شده اند. پس از اجرای دو سناریو، اندازه گیری عملکرد با استفاده از پروتکل‌های TCP و UDP انجام شده است. پس از ایجاد ترافیک، تاخیر و توان عملیاتی را به عنوان میانگین ده تکرار برای هر سناریو اندازه گیری کرده‌ایم. برای هر دو سناریوی معماری شبکه مبتنی بر شبکه نرم افزار محور و معماری مرسوم، همین رویه را دنبال شده و پارامترهای کیفیت سرویس اندازه گیری شده در طول اجرای سناریوها ثبت شده اند.

Pseudocode of the scenario

1. *Add controller: net.addController;*
   Set controller name to C0;
   Set controller ip address and port number;
   Set controller as remote with TCP connection;
2. *Add OpenFlow switches: net.addSwitch;*
   Set S1, S2 ,S3 as OVSKernelSWitches;
3. *Add Hosts: Net.addHost(h1, h2, h3, h4);*
   Set ip address for h1, h2, h3;
4. *Add links:*
   Net.addLink(h1, h2 to s1);
   Net.addLink(h3, h4 to s2);
   Net.addLink(h5, h6 to s3);
5. *Starting Controller;*
6. *Starting network with host and OVSSwitches;*
7. *Establish connection between hosts;*
8. *Establish client-server connection between h2 and h4;*
9. *Evaluate and record QoS parameters from the network;*

جدول ۱. شبه کد مربوط به تولید سناریو

برای ارزیابی عملکرد کیفیت سرویس سناریوها، توان عملیاتی و تأخیر دوعامل مهم هستند. توان عملیاتی، سرعت واقعی انتقال داده در شبکه و تأخیر، زمانی است که بسته برای رسیدن به مقصد از یک منبع صرف می‌کند.

یکی از چالش‌های پژوهش‌های مرتبط با شبکه تولید الگوی واقعی ترافیک شبکه است. برای اینکار نیاز به یک مولد ترافیک منعطف و سبک که بتواند انواع مختلف ترافیک را با پشتیبانی از رابط‌های مجازی تولید کند داریم. بسیاری از مولدهای ترافیک این شرایط را دارا نیستند. مثلا Netperf [۲۲] تنها قادر به تولید جریان‌های ثابت بوده ولی از نظر میزان مصرف پردازنده کارآمد بوده و از رابط‌های مجازی نیز پشتیبانی می‌کند. دیگر مولدهای جریان ترافیک شبکه مثل TRex[۲۳] و WARP۱۷ [۲۴] میزان مصرف CPU بیشتری داشته و همچنین از رابط های مجازی بطور محدود تر پشتیبانی می‌کنند.

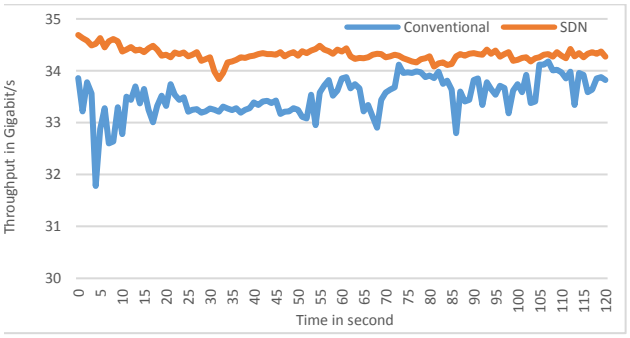
ما در این پژوهش، برای تولید ترافیک و اندازه گیری پهنای باند و توان عملیاتی، از ابزار iPerf [۲۵] استفاده می کنیم. iPerf یک ابزار کلاینت–سروری است که قادر به تولید ترافیک با پروتکل-های مختلف TCP و UDP و اندازه گیری توان عملیاتی و کیفیت ارتباط شبکه بین دو میزبان (مانند کلاینت و سرور) است. در این آزمایش ابتدا اسکریپت پایتون سناریوی پیشنهادی را اجرا کرده و سپس با استفاده از "xterm" به ترمینال میزبان ها دسترسی ایجاد می‌کنیم. برای برقراری ارتباط بین کلاینت و سرور، باید iPerf را با پارامترهای خاص روی هر شبیه سازی ترمینال کلاینت و سرور که توسط xterm باز می‌شود، اجرا کنیم. برای پروتکل TCP، در سمت سرور (host2)، iPerf با پارامتر "s" صادر شده که سمت سرور اتصال را تعیین می‌کند. پارامتر "p" پورت گوش دادن سرور را مشخص می‌کند. دستور کامل صادر شده " iPerf -s -p 6633 -i 1" است. به طور مشابه، در سمت کلاینت، دستور iPerf با پارامتر "c" اجرا شد که سمت مشتری اتصالات را تعیین می‌کند. پارامتر، "p" شماره پورت را تعیین می‌کند، "<server_ip_address>" آدرس IP سرور را تعیین می‌کند، و "t" زمان اجرای iPerf را به ثانیه تعیین می‌کند. دستور کامل صادر شده "Iperf –c <server_IP_address> -p 6633 –t 120" است. برای ایجاد یک اتصال UDP، باید از پارامتر "u" در سمت مشتری و سرور استفاده کنیم. اگرچه iPerf از هر دو پروتکل TCP و UDP به عنوان پروتکل انتقال پشتیبانی می‌کند، بین آزمایش‌های TCP و UDP در پهنای باند و نتایجی که خروجی iPerf بر می-گرداند تفاوت وجود دارد. در آزمایش با پروتکل TCP، فرستنده به اندازه حداکثر توان تحمل انتقال شبکه داده تولید می‌کند، در حالی که در آزمایش UDP نرخ انتقال را باید با پارامتر "b" تعیین کنیم، در غیر این صورت Mininet سرعت انتقال را روی یک مگابیت در ثانیه محدود می‌کند. در این آزمایش، نرخ انتقال را با پارامتر "b" روی ۴۰ مگابیت در ثانیه تنظیم کردیم. با اجرای iPerf در سمت سرور (host-2)، سرور شروع به گوش دادن به پورتی می‌کند که در دستور مشخص شده است. و همچنین با اجرای iPerf در سمت کلاینت (host-4) ترافیکی را در آن پورت با پارامترهای تعیین شده (نرخ انتقال خاص، مدت زمان انتقال و اندازه پنجره TCP) ایجاد می‌کند. با استفاده از iPerf ترافیک زمینه ای با نرخ ثابت ۱۰۰ بسته در ثانیه ایجاد کرده‌ایم. برای اندازه گیری تاخیر بین دو میزبان نیز می‌توانیم از پروتکل ICMP و ping استفاده کنیم.

سیستم عامل	Ubuntu 18
محیط شبیه ساز	Mininet 2.3.0d6
کنترل کننده	OpenFlow 1.3
مدت زمان آزمایش	120 seconds
اندازه پنجره TCP	85.3 Kbytes (standard)
اندازه بافر UDP	208 Kbytes
اندازه بسته ICMP	1500 bytes (1472+20+8, data, IP header, ICMP header)
تعداد کنترل کننده	1
تعداد سوییچ OpenFlow	3
تعداد میزبان	6
زمان انتظار برای اتصال جدید	1
فاصله بین بسته ها	280 us
پهنای باند لینک ها	40 mbps
ترافیک زمینه	100 mps

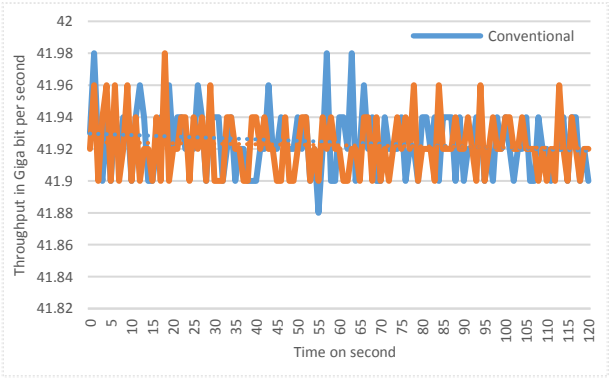
جدول ۲. تنظیمات مربوط به آزمایش

شکل ۴ توان عملیاتی به دست آمده جریان ترافیک پروتکل TCP در هر دو سناریوی مبتنی بر شبکه نرم افزار محور و سناریوی شبکه مرسوم را در طول ۱۲۰ ثانیه نشان می‌دهد. همانطور که مشاهده می‌شود میانگین توان عملیاتی به دست آمده توسط کنترل کننده شبکه نرم افزار محور حدود ۳۴,۳ گیگابیت بر ثانیه است در حالی

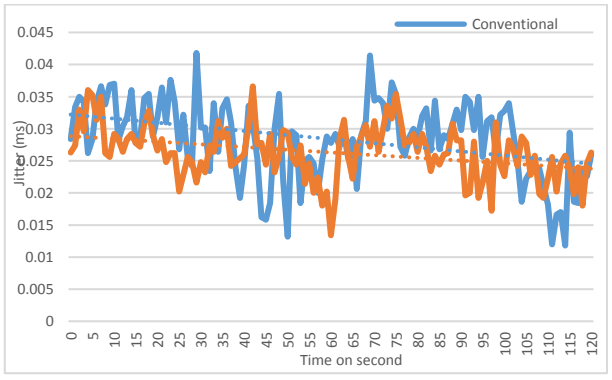
که میانگین توان عملیاتی به دست آمده توسط شبکه مرسوم حدود ۳۳,۴ گیگابیت بر ثانیه است. نتایج پروتکل UDP بسیار نزدیک است (همانطور که در شکل ۵ نشان داده شده است). شکل ۶ نشان می‌دهد که جیتر تحت شبکه مبتنی بر SDN کمتر است. چنانچه در نتایج نشان داده شده شبکه مبتنی بر شبکه نرم افزار محور به توان عملیاتی بالاتر تحت هر دو پروتکل TCP و UDP و همچنین جیتر بهتر تحت پروتکل UDP دست یافته است.



شکل ۴. پروتکل TCP: مقایسه توان عملیاتی

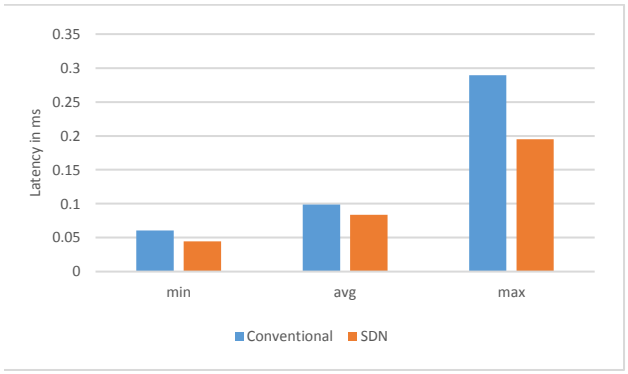


شکل ۵. پروتکل UDP: مقایسه توان عملیاتی



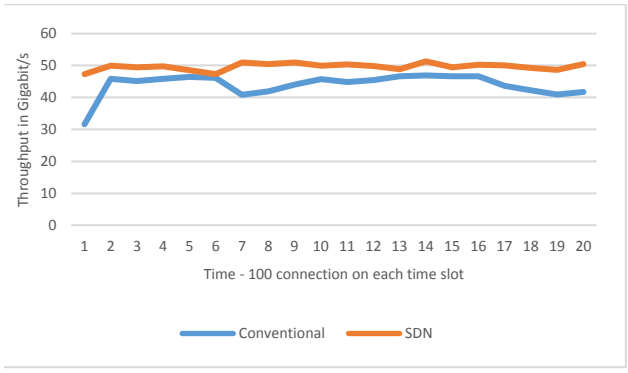
شکل ۶. مقایسه جیتر

مقایسه تاخیر بین شبکه‌های مبتنی بر شبکه نرم افزار محور و شبکه-های مرسوم در شکل ۷ نشان داده شده است. حداقل، میانگین و حداکثر تاخیر شبکه با اندازه بسته استاندارد ICMP اندازه گیری شده است. چنانچه مشاهده می‌شود سناریوی مبتنی بر شبکه نرم افزار محور در مقایسه با سناریوی شبکه مرسوم تاخیر کمتری دارد.



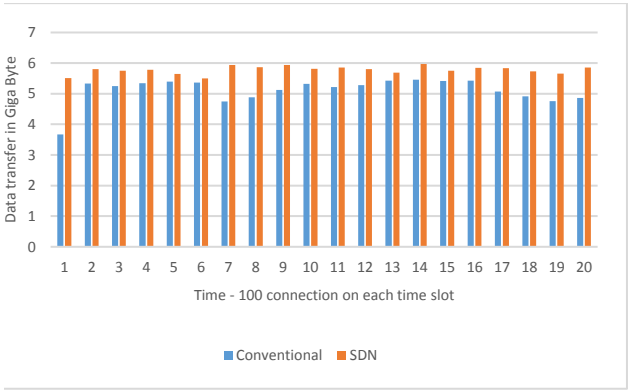
شکل ۷. مقایسه تاخیر

آزمایش با صد اتصال TCP موازی از کلاینت به سرور در هر دو سناریو تکرار شد. نتایج در شکل ۸ نشان داده شده است. همانطور که مشاهده می‌شود میانگین توان عملیاتی به دست آمده توسط سناریوی مبتنی بر شبکه نرم افزار محور ۴۹,۶۲ گیگابیت بر ثانیه و میانگین توان عملیاتی به دست آمده از سناریوی شبکه مرسوم ۴۳ گیگابیت بر ثانیه است.



شکل ۸. مقایسه توان عملیاتی با ۱۰۰ ارتباط موازی

همانطور که در شکل ۹ نشان داده شده است، میانگین حجم داده-های منتقل شده توسط سناریوی مبتنی بر شبکه نرم افزار محور، در طول آزمایش با ۱۰۰ اتصال ۵,۸ گیگابایت در ۲۰ ثانیه است در حالی که میانگین حجم داده‌های انتقال یافته توسط سناریوی شبکه مرسوم ۵,۱ گیگابایت است.



شکل ۹. مقایسه انتقال داده با ۱۰۰ اتصال

با توجه به نمودارها، مشاهده می‌شود که سناریوی شبکه مبتنی بر شبکه نرم افزار محور می‌تواند عملکرد بهتری در مقایسه با فناوری مرسوم شبکه داشته باشد. توجه به این نکته مهم است که ما سناریوها را با تعداد محدودی میزبان و سوئیچ ارزیابی کرده‌ایم و مقایسه بین این دو فناوری در مقیاس‌های بزرگتر نیاز به پژوهش بیشتری دارد. همچنین در سناریوی شبکه نرم افزار محور از کنترل کننده Ryu استفاده کردیم که مقایسه عملکرد بین کنترل کننده-های دیگر نیز می تواند موضوع پژوهش‌های بعدی باشد.

٦. نتیجه گیری

شبکه نرم افزار محور به عنوان یک فناوری نوظهور و رو به توسعه جنبه‌های جدید و تاثیرگذاری را برای شبکه‌های امروزی ایجاد کرده و از این رو یک فناوری امیدوارکننده برای بسیاری ازچالش‌ها است. این فناوری با توجه به قابلیت‌های برنامه ریزی، انعطاف پذیری و مدیریت متمرکز شبکه، ما را قادر می‌سازد تا مدیریت، نظارت، کنترل و بهبود پارامترهای کیفیت سرویس را در شبکه بهتر و موثرتر انجام دهیم.

در این مقاله، ما یک معماری شبکه مبتنی بر فناوری شبکه نرم افزار محور ارائه کرده و عملکرد آن را با رویکردهای مرسوم شبکه، مورد بررسی و تحلیل قرار دادیم. آزمایش‌ها برای ارزیابی توان عملیاتی، تأخیر و جیتر در دو سناریوی معماری شبکه مبتنی بر شبکه نرم افزار محور و معماری مرسوم شبکه انجام شد.

جریان‌های ترافیکی پروتکل‌های TCP و UDP تولید شده و عملکرد هر پروتکل در سناریوهای مربوطه اندازه گیری شد. علاوه بر این، ۱۰۰ اتصال TCP موازی تولید و عملکرد شبکه در هر دو سناریو به دست آمد. با توجه به عملکرد به دست‌آمده از آزمایش‌ها، می‌توان نتیجه گرفت که شبکه نرم افزار محور نسبت به شبکه‌های مرسوم عملکرد بهتری در توان عملیاتی و تأخیر دارد. آزمایش‌ها به ویژگی‌های سخت افزاری و توان پردازشی ماشین-های مجازی مانند CPU، RAM و غیره محدود شده است و واضح است که با بهبود برخی از محدودیت‌ها می‌توان شبکه‌های بزرگتر با پیکربندی‌های مختلف را در نظر گرفت. همچنین در سناریوی مبتنی بر شبکه نرم افزار محور استفاده از پروتکل‌های مختلف به جز Ryu برای کنترل کننده مرکزی ممکن است به نتایج متفاوتی دست یابد که نیاز به مطالعه و تحلیل دارد. علاوه بر مزیت‌های مورد پژوهش، این فناوری اجازه پیاده‌سازی الگوریتم‌های یادگیری ماشینی و بهینه سازی را در فرآیند زیر ساخت شبکه را می‌دهد. در این رابطه مسایلی از جمله موازنه بار کار سرویس دهنده‌ها، تصمیم گیری خودمختار و غیره توسط روش‌های مورد اشاره در بستر شبکه‌های نرم افزار محور می‌تواند موضوع پژوهش-های بعدی برای حل چالش‌های مختلف باشد.

مراجع (References)

[1] Cisco, "Cisco Annual Internet Report (2018–2023)," *White paper Cisco public*, 2018. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf (accessed Jan. 17, 2022).

[2] Open Networking Foundation (ONF), "Software-Defined Networking (SDN) Definition - Open Networking Foundation," *Open Networking Foundation*, 2018. https://opennetworking.org/sdn-definition/?nab=1&utm_referrer=https%3A%2F%2Fsdn.itrc.ac.ir%2F%3Fq%3Dfa%2Fcontent%2Fsdn (accessed Mar. 28, 2021).

[3] S. Dahmen-Lhuissier, "ETSI - Multi-access Edge Computing - Standards for MEC," *Etsi*, 2020. https://www.etsi.org/technologies/multi-access-edge-computing (accessed Jan. 28, 2022).

[4] A. S. Patel, P. Gandotra, R. Kumar, A. A. Kherani, B. Lall, and S. Mukherjee, "Standardization aspects of Tactile applications on Multi-access Edge Computing," *2022 14th Int. Conf. Commun. Syst. NETworkS*, pp. 899–904, Jan. 2022, doi: 10.1109/COMSNETS53615.2022.9668380.

[5] S. Javanmardi, M. Shojafar, R. Mohammadi, A. Nazari, V. Persico, and A. Pescapè, "FUPE: A security driven task scheduling approach for SDN-based IoT–Fog networks," *J. Inf. Secur. Appl.*, vol. 60, p. 102853, Aug. 2021, doi: 10.1016/J.JISA.2021.102853.

[6] A. Rahman *et al.*, "SDN–IoT empowered intelligent framework for industry 4.0 applications during COVID-19 pandemic," *Cluster Comput.*, vol. 3, pp. 1–18, Jul. 2021, doi: 10.1007/S10586-021-03367-4/TABLES/5.

[7] M. Moslehi, H. Ebrahimpor-Komleh, S. Goli, and R. Taji, "A QoS Optimization Technique with Deep Reinforcement Learning in SDN-Based IoT," *Majlesi J. Electr. Eng.*, vol. 15, no. 3, pp. 105–113, Sep. 2021, doi: 10.52547/mjee.15.3.105.

[8] T. Theodorou and L. Mamatas, "CORAL-SDN: A software-defined networking solution for the internet of things," in *2017 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN 2017*, Dec. 2017, vol. 2017-Janua, pp. 1–2, doi: 10.1109/NFV-SDN.2017.8169870.

[9] D. Sinh, L. V. Le, B. S. P. Lin, and L. P. Tung, "SDN/NFV - A new approach of deploying network infrastructure for IoT," in *2018 27th Wireless and Optical Communication Conference, WOCC 2018*, Jun. 2018, pp. 1–5, doi: 10.1109/WOCC.2018.8372689.

[10] I. Bedhief, M. Kassar, and T. Aguili, "From Evaluating to Enabling SDN for the Internet of Things," in *Proceedings of IEEE/ACS International Conference on Computer Systems and Applications, AICCSA*, Jan. 2019, vol. 2018-Novem, doi: 10.1109/AICCSA.2018.8612841.

[11] R. Muñoz *et al.*, "Integration of IoT, Transport SDN, and Edge/Cloud Computing for Dynamic Distribution of IoT Analytics and Efficient Use of Network Resources," *J. Light. Technol.*, vol. 36, no. 7, pp. 1420–1428, Apr. 2018, doi: 10.1109/JLT.2018.2800660.

[12] C. Xu, H. Lin, Y. Wu, X. Guo, and W. Lin, "An SDNFV-Based DDoS Defense Technology for Smart Cities," *IEEE Access*, vol. 7, pp. 137856–137874, 2019, doi: 10.1109/ACCESS.2019.2943146.

[13] C. Lin, Y. Bi, H. Zhao, Z. Liu, S. Jia, and J. Zhu, "DTE-SDN: A Dynamic Traffic Engineering Engine for Delay-Sensitive Transfer," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5240–5253, Dec. 2018, doi: 10.1109/JIOT.2018.2872439.

[14] S. Wang, J. Wu, S. Zhang, and K. Wang, "SSDS: A smart software-defined security mechanism for vehicle-to-grid using transfer learning," *IEEE Access*, vol. 6, pp. 63967–63975, Sep. 2018, doi: 10.1109/ACCESS.2018.2870955.

[15] S. Singh and R. K. Jha, "A Survey on Software Defined Networking: Architecture for Next Generation Network," *J. Netw. Syst. Manag.*, vol. 25, no. 2, pp. 321–374, Jan. 2020, doi: 10.1007/s10922-016-9393-9.

[16] A. Abuarqoub, "A review of the control plane scalability approaches in software defined networking," *Futur. Internet*, vol. 12, no. 3, p. 49, Mar. 2020, doi: 10.3390/fi12030049.

[17] A. Kumari and A. S. Sairam, "Controller placement problem in software-defined networking: A survey," *Networks*, vol. 78, no. 2, pp. 195–223, Sep. 2021, doi: 10.1002/NET.22016.

[18] A. Belkhiri and M. Dagenais, "Diagnostic and troubleshooting of OpenFlow-enabled switches using kernel and userspace traces," *Int. J. Commun. Syst.*, vol. 34, no. 14, p. e4920, Sep. 2021, doi: 10.1002/dac.4920.

[19] Y. Li, X. Guo, X. Pang, B. Peng, X. Li, and P. Zhang, "Performance Analysis of Floodlight and Ryu SDN Controllers under Mininet Simulator," *2020 IEEE/CIC Int. Conf. Commun. China, ICCC Work. 2020*, pp. 85–90, Aug. 2020, doi: 10.1109/ICCCWORKSHOPS49972.2020.9209935.

[20] "Apache License, Version 2.0." https://www.apache.org/licenses/LICENSE-2.0 (accessed Apr. 16, 2020).

[21] Z. Xiang and P. Seeling, "Mininet: an instant virtual network on your computer," *Comput. Commun. Networks*, pp. 219–230, Jan. 2020, doi: 10.1016/b978-0-12-820488-7.00025-6.

[22] "The Netperf Homepage." https://hewlettpackard.github.io/netperf/ (accessed Dec. 05, 2020).

[23] "cisco-system-traffic-generator/trex-core: trex-core site." https://github.com/cisco-system-traffic-generator/trex-core (accessed Dec. 05, 2020).

[24] "Juniper/warp17: The Stateful Traffic Generator for Layer 1 to Layer 7." https://github.com/Juniper/warp17 (accessed Dec. 05, 2020).

[25] "No Title." https://github.com/esnet/iperf (accessed Dec. 05, 2020).