

۲ پیاده‌سازی سوئیچ OpenFlow

در این فصل پیاده‌سازی سوئیچ OpenFlow (نسخه ۱.۰) را شرح خواهیم داد. همچنین سوئیچ‌های نرم افزاری و سخت افزاری مهم OpenFlow را معرفی خواهیم کرد. سپس به معرفی ابزار Mininet به عنوان محیطی یکپارچه برای بدست آوردن تجربه‌ای مطلوب در استفاده از سوئیچ‌ها و کنترلرهای OpenFlow می‌پردازیم. پیاده‌سازی پایه OpenFlow و محصولات سخت افزاری و نرم افزاری مرتبط با آن، در این فصل معرفی می‌گردد. علاوه بر آن، یک آزمایشگاه OpenFlow را با استفاده از نرم افزار مدل ساز شبکه Mininet بصورت قدم به قدم تشریح و تجربه می‌کنیم.

سوئیچ مرجع OpenFlow

سوئیچ OpenFlow یک عنصر پایه‌ای و اساسی برای ارسال بسته محسوب می‌گردد که از طریق اینترفیس و پروتکل OpenFlow در دسترس و قابل مدیریت است. اگرچه ممکن است چنین وضعیتی در نگاه اول ساده سازی سخت افزار سوئیچینگ در این معماری را تداعی کند، اما معماری‌های SDN مبتنی بر جریان، همچون OpenFlow، نیازمند مواردی همچون ورودی‌های جدول ارسال (Forwarding Table)، فضای بافر و شمارنده‌های آماری هستند که پیاده‌سازی آن در سوئیچ‌های قدیمی با ساخت افزار ASIC چندان ساده نیست. در یک شبکه مبتنی بر پروتکل OpenFlow، سوئیچ‌ها بر دو گونه‌اند: هایبریدی و محض. سوئیچ‌های هایبریدی علاوه بر اجرای عملکردها و پروتکلهای قدیمی (منظور سوئیچینگ لایه ۲ و لایه ۳)،

^۱ عبارت Application Specific Integrated Circuit، مداری مجتماع است که جهت اهدافی خاص طراحی شده است. از جمله مواردی که این نوع از مدارات مجتماع در آنها پیاده‌سازی شده‌اند، می‌توان به PDA‌ها، تجهیزات راهنمایی و رانندگی یا حتی اپلیکیشن‌های انتقال داده نام برد. در تجهیزات شبکه، این مدار مجتماع، برای پردازش عملیات سوئیچینگ بسته‌ها مورد استفاده قرار می‌گیرد. (متترجم)

از پروتکل OpenFlow نیز پشتیبانی می‌کنند. اما سوئیچ‌های OpenFlow محض، رویکرده متفاوت را در پیش گرفته‌اند. آنها مشخصه‌های سوئیچ‌های کلاسیک را در خود جای نداده‌اند و قادر کنترل داخلی (Onboard Control) هستند. این سوئیچ‌ها برای تصمیمات مربوط به ارسال کردن بسته، به یک کنترلر بیرونی کاملاً وابسته‌اند. اکثر سوئیچ‌های تجاری موجود، از نوع سوئیچ هایبریدی هستند.^۷ از آنجایی که سوئیچ‌های OpenFlow توسط یک اینترفیس باز (روی یک Session از نوع TLS مبتنی بر TCP) کنترل می‌شوند، در دسترس بودن و امن بودن این اینترفیس (اینترفیس OpenFlow) بسیار مهم است. از آنجاییکه این پروتکل، ارتباط بین سوئیچ OpenFlow و یک کنترلر OpenFlow را تعریف می‌کند، آن را می‌توان به عنوان یکی از پیاده‌سازیهای ممکن در تعاملات بین سوئیچ و کنترلر به حساب آورد (که این پروتکل را می‌توان یک پروتکل ارسال پیغام یا عبارتی Messaging Protocol دانست).

پیاده‌سازی مرجع سوئیچ OpenFlow، عبارت است از:

- سرویس `ofdatapath`، که جدول جریان را در فضای کاربر^۸ (User Space) اجرا می‌کند.
- برنامه `ofprotocol`، که کامپوننت کانال امن سوئیچ OpenFlow را اجرا می‌کند.
- ابزار `dpctl`، که جهت پیکربندی سوئیچ است.

توزیع کامپوننتهای سوئیچ OpenFlow شامل یک نرم افزار اضافی نیز می‌شود (برای مثال Controller، نام یک برنامه کنترلر ساده است که به هر تعداد از سوئیچهای OpenFlow متصل می‌شود. همچنین نرم افزار Wireshark که می‌تواند پروتکل OpenFlow را تحلیل کند). شکل زیر سوئیچ مرجع اینترفیس و سه گونه پیغام اصلی (کنترلر - به - سوئیچ، ناهمگام و متقارن) و زیر مجموعه‌های آنها را به تصویر می‌کشد. این پیغامها به طور مختصر در فصل پیشین توضیح داده شدند. در این بخش جزئیات مرتبط با اجرای پیغامها بیشتر معرفی می‌گردد. ارسال پیغامهای نوع کنترلر - به - سوئیچ، در ابتدا توسط کنترلر شروع می‌شوند، اما پس از آن سوئیچ OpenFlow می‌تواند تصمیم‌گیری کند که آیا به پیغامهای فوق پاسخ بدهد یا آنها را بدون پاسخ بگذارد.

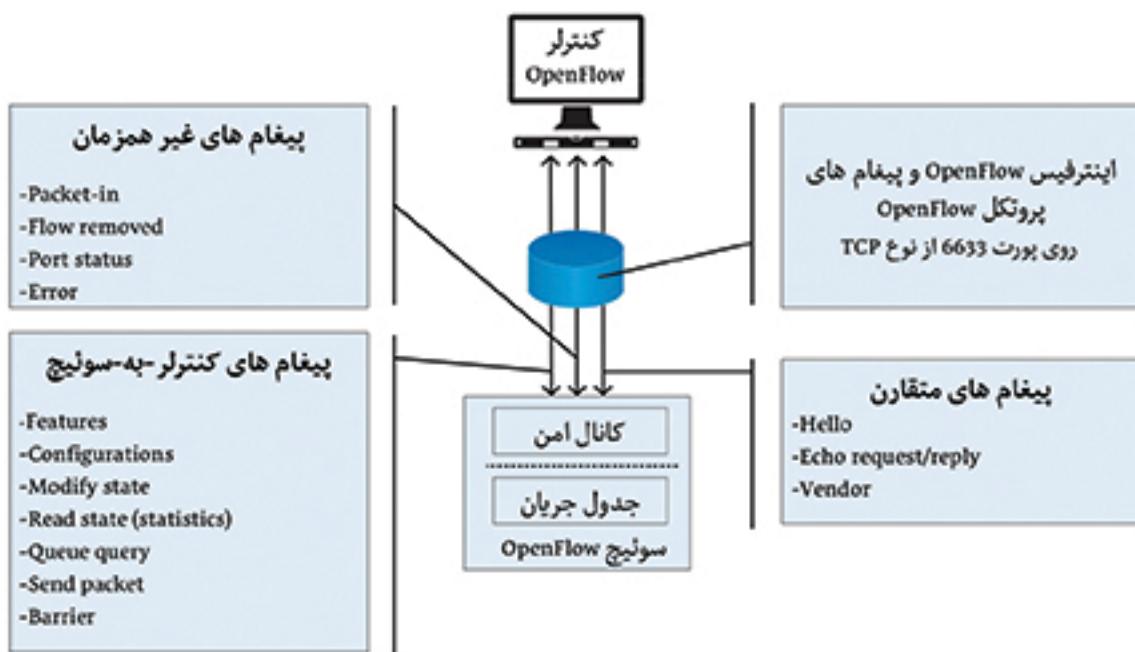
^۷ در زمان ترجمه این کتاب، شرکتهای مختلف در حوزه شبکه، محصولاتی از نوع OpenFlow محض را ارائه داده‌اند که از آن جمله می‌توان به محصول شرکت NEC اشاره کرد. (متترجم)

^۸ فضای کاربر با User Space را می‌توان بخشی از حافظه سیستم بر شمرد که کدهایی را که خارج از کرنل سیستم عامل در حال اجرا شدن هستند را در خود دارد. (متترجم)

پروتکل OpenFlow نسخه 1.3.0 امکان پشتیبانی اختیاری را برای ارتباطات رمزگاری شده TLS و تبادل گواهی بین سوئیچ (ها) و کنترلر (های) OpenFlow، فراهم می‌کند. با این حال، شکل دقیق پیاده‌سازی و گواهی هنوز مشخص نشده است. به علاوه، ویژگی‌های امنیتی دقیق و درستی در خصوص طرحهایی که از قابلیتهای چند کنترلری بودن OpenFlow، بهره می‌برد، در حال حاضر خارج از قلمرو مشخصه‌های OpenFlow است. تا به حال روش ویژه‌ای وجود نداشته که تنها اجازه دسترسی محدود و جزئی را به یک کنترلر مجاز OpenFlow بدهد. همچنین به باد داشته باشد که در این کتاب متنها به خصوصیات OpenFlow نسخه 1.0.0 خواهیم پرداخت. پیاده‌سازی مرجع OpenFlow نسخه 1.0.0 را می‌توانید از آدرس www.openflow.org/wp/downloads/ دریافت کنید.



۲-۱
انواع پیغام‌ها در
پروتکل OpenFlow



پیغام‌های کنترلر - به - سوئیچ برای مدیریت مستقیم سوئیچ یا مشاهده وضعیت سوئیچ بکار می‌روند. در ادامه به معرفی این پیغام‌ها خواهیم پرداخت:

خصوصیات (Features): به محض برقراری یک TLS Session (برای مثال یک TLS Session از نوع TCP، روی پورت 6633)، کنترلر یک پیغام `OFPT_FEATURES_REQUEST` به سوئیچ ارسال کرده و سوئیچ OpenFlow با فرستادن خصوصیات و قابلیتهای موجود و قابل پشتیبانی (از طریق پیغام `OFPT_FEATURES_REPLY`)، به این درخواست پاسخ می‌دهد. از جمله خصوصیاتی که سوئیچ

OpenFlow در پاسخ به درخواست `OFPT_FEATURES_REQUEST`، به سمت کنترلر باز می‌گرداند، می‌توان شناسه Datapath یعنی `Datapath_id`، تعداد جدولهای جریان پشتیبانی شده بوسیله Datapath (سوئیچ OpenFlow)، قابلیت‌های سوئیچ، Action‌های پشتیبانی شده و تعاریف پورتها را نام برد. فیلد `Datapath_id` فیلدهای این سوئیچ OpenFlow (که ما آن را Datapath هم می‌نامیم) یک شناسه انحصاری دریافت می‌کند. این عنصر ۶۴ بیت است که ۴۸ بیت کم ارزش‌تر آن را آدرس MAC سوئیچ تشکیل داده و ۱۶ بیت با ارزش‌تر، متعلق به سازنده است.

پیکربندی Configuration: کنترلر به ترتیب با پیغامهای `OFPT_SET_CONFIG` و `OFPT_GET_REPLY` قادر به تنظیم و ارسال درخواست در خصوص پارامترهای پیکربندی در سوئیچ است. سوئیچ از طریق پیغام `OFPT_GET_CONFIG_REPLY` به درخواست پیکربندی پاسخ می‌دهد، ولی پس از اعمال دستور تنظیم پیکربندی (یعنی `OFPT_SET_CONFIG`)، پاسخی باز نمی‌گردد.

وضعیت اصلاح Modify State: اصلاح جدول جریان از جانب کنترلر، با پیغام `OFPT_FLOW_MOD` انجام می‌شود و اصلاح رفتار فیزیکی پورتها با پیغام `OFPT_PORT_MOD` صورت می‌پذیرد. از جمله فرمانهای اصلاح جریان می‌توان به `DELETE`, `MODIFY_STRICT`, `MODIFY_ADD` و `DELETE_STRICT` اشاره کرد که در فصل ۱ به آنها پرداخته شد. بیتهای Port Configuration نشان می‌دهند که آیا یک پورت توسط مدیر شبکه خاموش شده است یا خیر. همچنین اختیاراتی برای بررسی بسته‌های مربوط به پروتکل STP و نحوه رسیدگی به بسته‌های دریافتی و خروجی در سوئیچ را مشخص می‌کند. کنترلر برای فعال کردن پروتکل STP روی یک پورت می‌تواند فرمان `OFPFFL_NO_STP` را با مقدار ۱ تنظیم کرده یا برای غیر فعال کردن آن بر روی پورت، مقدار آن را با مقدار ۰ تنظیم کند. پیاده‌سازی مرجع OpenFlow این بیت را بصورت پیشفرض صفر (وضعیت STP فعال) تنظیم می‌کند.

حالت خواندن Read State: کنترلر می‌تواند وضعیت سوئیچ را با استفاده از پیغام `OFPT_STATS_REPLY` پرس و جو کند. سوئیچ با یک یا چند پیغام `OFPT_STAT_REQUEST` پاسخ می‌دهد. فیلدهای تحت عنوان `type` در تبادل این پیغامها وجود دارد که نوع اطلاعات در حال تبادل را مشخص کرده (توصیف سوئیچ OpenFlow، آمارهایی همچون جریان منحصر بفرد، مجموع جریان، جدول جریان، پورت فیزیکی، صفاتی برای یک پورت و پیغامهای مربوط به خدمات دهنده‌ای خاص `Vendor-specific`) و تعیین می‌کند که فیلد، چگونه تفسیر شود.

پرس و جوی صف Queue Query: سوئیچ OpenFlow از قابلیت ^۷ QoS پشتیبانی می‌کند، اما این پشتیبانی محدود بوده و تنها از طریق یک ساز و کار صف بندی ساده، فراهم می‌شود. یک پورت

می‌تواند از یک یا چند صفت، جهت نگاشت جریانها بر روی خود و در نهایت اعمال QoS بهره برد. جریانهایی که برای یک صفت خاص مشخص شده‌اند، براساس پیکربندی منحصر بفرد همان صفت با آنها رفتار می‌شود (برای مثال، کنترل با حداقل نرخ انتقال). به یاد داشته باشید که پیکربندی صفت، خارج از پروتکل OpenFlow (برای مثال، از طریق رابط خط فرمان)، یا پروتکل پیکربندی بیرونی دیگری، صورت می‌پذیرد. کنترلر می‌تواند با استفاده از پیغام Queue Query، در خصوص صفت پیکربندی شده بر روی یک پورت خاص، از سوئیچ پرس و جو کند.

ارسال بسته (Send Packet): با استفاده از این پیغام که `OFPT_PACKET_OUT` نام دارد، کنترلر قادر است بسته‌ها را از یک پورت خاص سوئیچ OpenFlow به بیرون ارسال کند.

مانع (Barrier): این پیغام زمانی ارسال می‌شود که کنترلر می‌خواهد اطمینان حاصل کند که به وابستگی‌های (Dependencies) پیغام، توجه می‌شود یا برای عملیات‌های کامل شده، پیام اطلاع‌رسانی (Notification) دریافت کند. پیغام مورد نظر `OFPT_BARRIER_REQUEST` است. این پیغام محتوایی ندارد و تنها به سوئیچ دستور می‌دهد که قبل از آنکه به سراغ پردازش‌های جدید برود، پردازش‌های گذشته را که هنوز به اتمام نرسیده‌اند، به پایان برساند. سوئیچ OpenFlow براساس اولویت دریافت بسته، باید پردازش تمام پیغام‌های از پیش دریافت شده را قبل از اجرای هر پیامی که بعد از پیغام Barrier Request باشد، به اتمام برساند. در این وضعیت هر پیغامی که بعد از پیغام Barrier Request توسط سوئیچ دریافت می‌شود، تا تمام شدن پردازش‌های قبلی، داخل صفت قرار می‌گیرد. وقتی پردازش جاری کامل شد، سوئیچ باید یک پیغام `OFPT_BARRIER_REPLY` با شناسه تراکنش (`xid`) مربوط به Barrier Request ارسال کند.

پیغام‌های ناهمزمان (Asynchronous Messages)

پیغام‌های ناهمزمان توسط سوئیچ ایجاد و ارسال می‌شوند. سوئیچ از این نوع پیغام‌ها برای مطلع کردن و بروز کردن کنترلر در رابطه با رویدادهای شبکه و تغییرات حالت خود، استفاده می‌کند. سوئیچها پیغام‌های ناهمزمان را برای اعلام رسیدن بسته، حذف جریان از جدول جریان، تغییر وضعیت پورت و یا وقوع یک خطا به کنترلر ارسال می‌کنند.

زمانیکه بسته‌ها توسط سوئیچ OpenFlow دریافت می‌شوند، سوئیچ در ابتدا آنها را بافر کرده و سپس با استفاده از پیغام `OFPT_PACKET_IN`، آن بسته‌ها را به کنترلر می‌فرستد. وقتی بسته‌ای در سوئیچ بافر می‌شود، بخشی از بدنه پیغام `OFPT_PACKET_IN`، حاوی اطلاعاتی از بسته خواهد بود، که در بخش داده (Data) پیغام قرار داده می‌شود. اگر بسته به دلیل وجود یک Action معادل Send-to-Controller، به کنترلر فرستاده شود، بایتهای `max_len` ارسال می‌شوند و اگر بسته به دلیل نبود

وروودی جریان در جدول جریان، به کنترلر ارسال شود، تنها بایتهای `miss_send_len` به سمت کنترلر فرستاده می‌شوند. اگر بسته در درون سوئیچ بافر نشود، تمام بسته در بخش داده پیغام اضافه شده و به کنترلر ارسال می‌شود. سوئیچهایی که کار پیاده‌سازی بافر را انجام می‌دهند باید اندازه بسته‌هایی که بافر می‌شوند را تعیین کنند. همچنین این سوئیچها لازم است طول مدت زمان بافر شدن آن بسته‌ها را نیز قبل از استفاده مجدد از این قابلیت، مشخص کنند. یکی از مواردی که یک سوئیچ بایستی به دقت آن را مورد بررسی قرار دهد، وضعیتی است که یک پیغام بافر شده از نوع `packet_in`، هیچ پاسخی از کنترلر دریافت نکند. هنگامیکه زمان ماندگاری جریانها در جدول جریان سوئیچ به پایان می‌رسد (Flows Timeout)، سوئیچ OpenFlow این تغییر را با پیغام `OFPT_FLOW_REMOVED` به کنترلر اطلاع می‌دهد (البته در صورتیکه کنترلر، درخواستی مبنی بر مطلع شدن از تغییرات سوئیچ، داده باشد). فیلدی‌های `duration_nsec` و `duration_sec` درون پیغام `OFPT_FLOW_REMOVED`، نشان دهنده زمان سپری شده از لحظه نصب جریان در سوئیچ هستند. کل مدت زمان را می‌توان با فرمول $\text{duration_sec} \times 10^9 + \text{duration_nanosec}$ ، به نانو ثانیه `idle_timeout` محاسبه کرد. اما پیشنهاد می‌شود پیاده‌سازیها با دقت میلی ثانیه ارائه شوند. فیلد مستقیماً از `FLOW_MOD` که وروودی جدول جریان را ایجاد می‌کند، استخراج می‌شود. وقتی پورتهای فیزیکی یک سوئیچ، اضافه، حذف و یا اصلاح شوند، لازم است کنترلر با استفاده از پیغام `OFPT_PORT_STATUS` از این تغییر مطلع شود. موقعی وجود دارد که سوئیچ دچار خطا یا اختلال می‌شود. در این موارد سوئیچ OpenFlow باید مشکل را به کنترلر گزارش کند. پیغام فوق مشتمل بر نوع خطا (Error Type)، کد خطا (Error Code) و یک داده با طول متغیر بوده که باید برآسانس نوع و کد خطا تفسیر شود. در بیشتر موارد، بخش داده، متن خطا را نمایش می‌دهد. شش گونه خطا در این حالت وجود دارد. خطای `OFPET_HELLO_FAILED` که نشان می‌دهد، پروتکل Hello دچار مشکل شده است. خطای `OFPET_BAD_REQUEST` به موردی اشاره می‌کند که درخواست دریافت شده توسط کنترلر، نامفهوم بوده است. اگر توصیف Action دچار خطا شود، پیغام خطای `OFPET_BAD_ACTION` صادر می‌شود. اگر درخواستهای `PORT_MOD` یا `FLOW_MOD` دچار مشکل شوند، نوع خطا به ترتیب `OFPET_PORT_MOD_FAILED` و `OFPET_FLOW_MOD_FAILED` خواهد بود. مشکلات در عملیات صف پورت با پیغام خطای `OFPT_QUEUE_OP_FAILED` دسته بندی می‌شود.

پیغامهای متقارن (Symmetric Messages)

پیغام Hello در قالب `OFPT_HELLO`، پیغامهای Echo Reply و Echo Request و پیغام `Vendor` جزوی از پیغامهای متقارن OpenFlow محسوب می‌شوند. در پیاده‌سازی مرجع OpenFlow که شامل یک پردازش فضای کاربر (User Space Proces) و ماژول کرزل است، پیغامهای Echo Request و

Echo Reply در کریل اجرا می‌شوند. این نوع پیاده‌سازی باعث می‌شود محاسبه تأخیر (Latency) بین دو نقطه از شبکه، بطور دقیق تری انجام شود. فیلد Vendor در پیغام `OFPT_VENDOR` اندازه ۲۲ بیتی دارد که منحصرًا مازنده و یا شرکت خاصی را مشخص می‌کند. اگر با ارزشترین بایت صفر باشد، سه بایت بعدی (یعنی ۲۴ بیت)، IEEE OUI متعلق به مازنده هستند. اگر یک سوئیچ، شناسه ۲۲ بیتی یک مازنده محصول را شناسایی نکند، باید یک پیغام `OFPT_ERROR` همراه با یک نوع خطای `OPPBRC_BAD_VENDOR` و یک کد خطای `OFPT_BAD_REQUEST` ارسال کند.

پیاده‌سازی‌های ساخت افزاری

نسخه مرجع OpenFlow (یعنی OpenFlow نسخه ۱.۰.۰)، SDN اصلی و اولیه است و تکنولوژیهایی را فعال می‌کند که در حال حاضر در ساخت افزارهای ارزان قیمت حوزه شبکه^۵ تعبیه شده‌اند. در این بخش قصد نداریم سوئیچهای نوع OpenFlow-enabled و تولیدکنندگان آن را به طور کامل و دقیق توضیح دهیم. ما ترجیحاً فهرستی کلی از چند گزینه را که در بازار موجود است ارائه خواهیم داد. جدول زیر فهرستی از سوئیچهای تجاری را که در حال حاضر موجود هستند، به همراه تولیدکنندگانشان و نسخه OpenFlow پیاده‌سازی شده در آنها را نشان می‌دهد.

جدول ۲-۱

شرکت‌های تولید
کننده سوئیچ‌های
هایبریدی*

Manufacturer	Switch Models	OpenFlow Version
Brocade	NetIron CES 2000 Series, CER 2000,	1.0
Hewlett Packard	3500/3500yl, 5400zl, 6200zl, 6600, 8200zl	1.0
IBM	RackSwitch G8264, G8264T	1.0
Juniper	EX9200 Programmable Switch	1.0
NEC	PF5240, PF5820	1.0
Pica8	P-3290, P-3295, P-3780, P3920	1.2
Pronto	3290 and 3780	1.0
Broadcom	BCM56846	1.0
Extreme Networks	BlackDiamond 8K, Summit X440, X460, X480	1.0
Netgear	GSM7352Sv2	1.0
Arista	7150, 7500, 7050 series	1.0

^۵ تجهیزات ارزان قیمت حوزه شبکه (Commodity-networking Hardware) به دستگاه یا کامپوننتی گلته می‌شود که ارزان قیمت بوده و بصورت گسترده در دسترس است. امروزه برخی طراحان در حوزه Big Data از این نوع تجهیزات بهره می‌برند. (مترجم)
* در زمان ترجمه این کتاب، شرکتهای دیگری همچون Cisco با ارائه تجهیزات همچون Nexus 3000، Nexus 9300، Nexus 31128PQ، سری ASR، ISR و برخی محصولات Catalyst از این پرتوکل پشتیبانی می‌کنند. از دیگر شرکتهایی که در این حوزه فعال هستند می‌توان Big Switch Networks، Infoblox، ADVA، Infinera، Extreme Networks و ... را نام برد. (مترجم)

سوئیچ‌های مبتنی بر نرم افزار

در حال حاضر چندین سوئیچ نرم افزاری OpenFlow وجود دارد که کاربردهای مختلفی دارند. از نمونه‌های این کاربردها می‌توان به اجرای پست‌رسانی بر روی OpenFlow یا توسعه و آزمایش اپلیکیشن‌های شبکه مبتنی بر OpenFlow، اشاره نمود. فهرستی از سوئیچ‌های نرم افزاری کنونی، همراه توضیحات مختصری همچون زبان پیکربندی و نسخه OpenFlow که استفاده کردند، در ادامه خواهد آمد.

- **Open vSwitch:** یک سوئیچ مجازی چند لایه و در سطح تجاری است که تحت لیسانس Apache نسخه 2.0 قرار دارد. این سوئیچ از طریق توسعه در حوزه برنامه نویسی، جهت اضافه کردن ویژگی اتوماتیک‌سازی شبکه، طراحی شده است. همچنین از طریق این توسعه، توانایی پشتیبانی از اینترفیسها و پروتکلهای استاندارد مدیریتی (مانند، NetFlow، sFlow، OVSDB، OpenFlow و غیره) را دارد.
- **Indigo:** یک پیاده‌سازی متن باز از پروتکل OpenFlow است که بر روی سوئیچ‌های فیزیکی اجرا می‌شود و از مشخصه‌های سخت افزاری سوئیچ‌های اترنت مدل ASIC برای اجرای OpenFlow در نرخ های خط^۷ (Line Rate) استفاده می‌کند. این سوئیچ مبتنی بر پیکربندی OpenFlow مرجع است.
- **Pantou (OpenWRT):** ابزاری است که می‌تواند یک مسیریاب بی‌سیم یا یک Access Point را به یک سوئیچ هایبریدی تبدیل کند. پروتکل OpenFlow به عنوان یک اپلیکیشن بر روی سیستم عامل OpenWRT پیاده‌سازی شده است. ابزار Pantou مبتنی بر نسخه منتشر شده BackFire OpenWrt (Linux 2.6.32) که آن را مورد استفاده قرار داده است، همان نسخه مرجع دانشگاه استنفورد (Userspace) است. در زمان ترجمه این کتاب، این سوئیچ از محصول Broadcom و برخی مدل‌های Access Point از شرکتهای TP-LINK و LinkSys با چیپست‌های Atheros پشتیبانی می‌کند.
- **LINC:** یک پروژه متن باز است که با تلاش شرکت FlowForwarding هدایت می‌شود و پیاده‌سازی آن تحت لیسانس Apache 2.0، براساس پروتکل OpenFlow نسخه 1.2 و 1.3.1 می‌شود.

⁷ نرخ خط و با Line Rate به توان انتقال تراولیک داده توسط سوئیچ، مسیریاب، فایروال و باحت مدارهای مجتمع ASIC با دستگاه شبکه ای اطلاق می‌شود. به عنوان مثال یک ASIC که ۱۲ اینترفیس با سرعت ۱Gbps را داشته باشد، نرخ خطی معادل ۱۲Gbps را دارا است. به زبان ساده‌تر اینطور می‌توان بیان کرد که نرخ خط، به پهنای یاند در لایه فیزیکی، در درون یک دستگاه شبکه، اطلاق می‌شود. (ترجم)

صورت پذیرفته است. هدف از طراحی سوئیچ LINC، پیاده‌سازی آن بر روی سخت افزارهای موجود و ارزان قیمت سری x86 بوده است. این سوئیچ بر روی پلتفرم‌های گوناگونی که توان Mac، Windows، Solaris، Linux، Erlang را دارند، مانند سیستم عامل‌های FreeBSD و OS قابل اجراءست.

Of13softswitch •
یک سوئیچ نرم افزاری فضای - کاربر (Userspace) بوده که با پروتکل Ericsson TrafficLab OpenFlow نسخه ۱.۳ سازگاری دارد و مبتنی بر سوئیچ نرم افزاری ۱.۱ است. آخرین نسخه این سوئیچ نرم افزاری شامل بخش پیاده‌سازی سوئیچ (ofdatapath)، کانالی امن برای اتصال سوئیچ به کنترلر (ofprotocol)، یک کتابخانه برای مکالمه با OpenFlow نسخه ۱.۳ (oflib) و یک ابزار پیکربندی (dpctl) است. این پروژه توسط مرکز تحقیقات اریکسون (Ericsson Innovation Center) در برزیل پشتیبانی می‌شود. اما نگهداری آن بر عهده موسسه CPqD^۸ با همکاری مرکز تحقیقات اریکسون در حوزه فنی است.

آزمایشگاه Mininet با OpenFlow

Mininet ابزاری نرم افزاری است که اجازه می‌دهد کل شبکه مبتنی بر OpenFlow، بر روی یک کامپیوتر مدل سازی (Emulate) شود. Mininet از مجازی‌سازی مبتنی بر پردازش سبک (فضای نامهای شبکه لینوکس^۹ و معماری Container لینوکس^{۱۰}) برای پیاده‌سازی تعداد زیادی میزبان و سوئیچ (برای نمونه ۴۰۹۶ عدد) بر روی یک کرنل سیستم عامل، استفاده می‌کند. نرم افزار Mininet می‌تواند کرنل یا سوئیچهای OpenFlow فضای کاربر، کنترلرهایی برای مدیریت سوئیچ‌ها و میزبانهایی برای برقراری ارتباط در شبکه مدل شده، ایجاد کند. Mininet با استفاده از کارت‌های شبکه مجازی که با "Veth" نام گذاری شده‌اند، میزبانها و سوئیچ‌ها را به هم متصل می‌کند. Mininet توسعه اولیه، اشکال زدایی (Debugging)، آزمایش و فرایند گسترش یک شبکه مبتنی بر پروتکل OpenFlow را بطور قابل توجهی ساده می‌کند. اپلیکیشن‌های جدید شبکه را می‌توان در ابتدا بر روی شبکه‌های مدل شده‌ای که درون Mininet ایجاد شده‌اند، آزمایش کرده و بهبود بخشید. سپس می‌توان آنها را به

^۸ موسسه Centro de Pesquisa e Desenvolvimento(CPqD)، سازمانی مستقل بوده که هدف اصلی آن، افزایش توان رقابتی کشور برزیل در حوزه فناوری اطلاعات در بین کشورهای آمریکای لاتین است. این سازمان با ورود به حوزه‌های امنیت، صنعت، مالی و سیستمهای چند رسانه‌ای توافته نقش استراتژیک برای خود ایجاد کند. (ترجم)

^۹ Linux Network Namespace

^{۱۰} Linux Container Architecture

^{۱۱} Virtual Ethernet

زیرساخت واقعی انتقال داد. بصورت پیش‌فرض، Mininet از پروتکل OpenFlow نسخه 1.0 پشتیبانی می‌کند.^{۲۲} با این حال ممکن است برای پشتیبانی از سوئیچ نرم افزاری که نسخه جدیدتری از OpenFlow را پیاده‌سازی می‌کند، اصلاح شود. برخی از خصوصیات کلیدی و مزایای Mininet به شرح زیر است:

- Mininet، شبکه‌ای از میزبانها، سوئیچ‌ها، کنترلرها و لینکهای مجازی ایجاد می‌کند.
 - میزبانهای Mininet، نرم افزار شبکه لینوکس را اجرا می‌کنند و سوئیچ‌هایی از OpenFlow پشتیبانی می‌کنند. Mininet را می‌توان یک آزمایشگاه OpenFlow ارزان قیمت برای بهبود و گسترش اپلیکیشن‌های OpenFlow در نظر گرفت. Mininet پیاده‌سازی و آزمایش توپولوژیهای پیچیده را بدون نیاز به کابل کشی یک شبکه فیزیکی محقق می‌سازد.
 - دارای یک محیط مبتنی بر خط فرمان (CLI) است که آگاه به وجود توپولوژی و آگاه به وجود OpenFlow بوده و برای اشکال زدایی یا اجرای آزمایش‌های گسترده شبکه، بکار می‌رود.
 - شما می‌توانید ابتدا به ساکن و بدون هیچ برنامه‌نویسی، شروع به استفاده از Mininet کنید و در عین حال Mininet می‌تواند API Python آسان و توسعه‌پذیری را برای ایجاد شبکه و آزمایش آن فراهم کند.
 - بجای این که ابزاری برای شبیه‌سازی باشد، یک محیط برای مدل کردن شبکه محسوب می‌شود که کدهای واقعی و اصلاح نشده را اجرا می‌کند؛ از جمله کد اپلیکیشن، کد کرنل سیستم عامل و کد Control Plane (مریوط به هر دو کد کنترلر سوئیچ OpenFlow و کد سوئیچ Open vSwitch)
 - نصب Mininet آسان بوده و بصورت یک فایل Image در قالب یک ماشین مجازی از پیش آماده در دسترس است. این فایل در خود، ابزارهای از پیش نصب شده مریوط به OpenFlow نسخه 1.0 را داشته و بر روی نرم افزار VMware یا VirtualBox در محیط سیستم عامل‌های لینوکس، ویندوز و مک قابل اجرا است.
- در ادامه این فصل ما آموزش کلی کار با Mininet را ارائه می‌کنیم و در فصول بعدی این کتاب نیز این مباحث مورد استفاده قرار خواهند گرفت.

^{۲۲} در هنگام ترجمه این کتاب، نرم افزار Mininet نسخه 2.2.1 ارائه شده است که از OpenFlow نسخه 1.3 پشتیبانی می‌کند. (ترجم)

شروع کار با Mininet

آسان ترین راه برای شروع کار با نرم افزار Mininet، دانلود یک فایل از پیش آماده ماشین مجازی Mininet بوده که شامل یک سیستم عامل Ubuntu و ابزارهای مورد نیاز آن است. این ماشین مجازی شامل کدهای باینتری OpenFlow، ابزارهای از پیش نصب شده برای پشتیبانی از شبکه های بزرگ، و همچنین خود Mininet است. علاوه بر نصب ماشین مجازی از پیش آماده، علاقمندان این مبحث می توانند Mininet را بصورت ساده از کد منبع (Source Code) یا بسته های نرم افزاری موجود در Ubuntu نصب کنند.

[مثالهای این فصل براساس Mininet نسخه 2.0 ارائه شده است. آخرین نسخه Mininet را می توانید از آدرس زیر دانلود کنید.^{۱۴} 

www.mininet.org/download

در صورتی که بخواهید از فایل ماشین مجازی (فایلی با پسوند ^{۱۵} OVF) استفاده کنید، باید یک نرم افزار مجازی ساز را دانلود کرده و بر روی کامپیوترتان نصب کنید. نرم افزار VirtualBox (رایگان و تحت لیسانس GPL) یا VMware Player (رایگان برای استفاده غیر تجاری)، از جمله نرم افزارهای مجازی ساز رایگان موجود هستند و بر روی سیستم عاملهای ویندوز، لینوکس و OS X قابل نصب و استفاده هستند. Mininet یک فایل OVF تقریبا 1 گیگابایتی است که می تواند توسط VMware یا VirtualBox اجرا شود. برای استفاده از فایل Image ماشین مجازی در محیط VirtualBox، وارد منوی Player شده و گزینه Import Appliance را انتخاب کنید. با توجه به اینکه ماشین مجازی Mininet نیاز به دو کارت شبکه مجازی دارد، برای افزودن دومین کارت شبکه، به بخش Settings رفته و گزینه host-only network adapter را انتخاب کنید. این کارت شبکه جهت اتصال به ماشین مجازی مورد استفاده قرار می گیرد. اگر از نرم افزار VMware استفاده می کنید، به بخش File رفته، منوی Open را باز کرده و

^{۱۴} در هنگام ترجمه این کتاب، آخرین نسخه منتشر شده از Mininet نسخه 2.2.1 است. از جمله ویژگیهای این نسخه می توان به این موارد اشاره کرد: ۱. پشتیبانی از نسخه 1.3 پروتکل OpenFlow. ۲. اضافه کردن گره های جدید شامل OVSBridge و LinuxRouter و LinuxBridge نسخه های قبل (متترجم)

^{۱۵} مفهوم Open Virtualization Format (OVF) یک استاندارد باز است که برای پسته بندی و توزیع Appliance های مجازی و با نرم افزارهایی که می توانند روی ماشین مجازی اجرا شوند، مورد استفاده قرار می گیرد. فایلهایی که پسوند OVF را پذک می کشند، اغلب سیستم عاملهای از پیش پیکربندی شده ای هستند که برای اهداف خاص ایجاد شده اند و اگر آنها را به داخل محیط مجازی Import کنید، می توانید برآختن آنها اجرا کرده و مورد استفاده قرار دهید. (متترجم)

^{۱۶} سیستم عاملهای OS X، مجموعه ای از سیستم عاملهای گرافیک مبتنی بر یونیکس است که توسط شرکت اپل توسعه بالته و بازاریابی می شوند. این سیستم عاملها به منظور اجرا بر روی کامپیوترهای مکینتاش طراحی شده اند. (متترجم)

فایل OVF خود را انتخاب کنید. همچنین برای افزودن کارت شبکه دوم، گزینه *Edit virtual machine setting* را انتخاب کرده و کارت شبکه مورد نظر خود را انتخاب کنید. ممکن است از شما خواسته شود تا VMware Tools را روی ماشین مجازی نصب کنید. در صورت مشاهده چنین درخواستی آن را نادیده بگیرید. در این کتاب ما VMware را به عنوان سیستم مجازی ساز، جهت کار با Mininet انتخاب کردیم.

پس از آنکه فایل Image را به داخل نرم افزار مجازی ساز منتقل کردیم، حال شروع به کار با Mininet می‌کنیم. در ادامه، گامهای اولیه برای شروع کار با این نرم افزار شرح داده می‌شوند:

- فایل ماشین مجازی Miminet را در برنامه *Miminet* ساز انتخابی خود وارد کرده و راه اندازی کنید. (*VMware Player*) در اسکرین شاتها نشان داده شده است).
- با استفاده از نام کاربری و رمز ورود پیش فرض، وارد ماشین مجازی Mininet شوید. نام کاربری و همچنین رمز ورود بصورت پیش فرض *mininet* هستند. در نظر داشته باشید که نمی‌توانید با حساب کاربری *root* وارد شوید. می‌توانید از فرمان *sudo* برای اجرای یک فرمان با اختیارات خاص کاربر ویژه، استفاده کنید.
- برای برقراری یک اتصال SSH به ماشین مجازی Mininet، باید آدرس IP ماشین مجازی را بیابید. این آدرس در *VMware Player* احتمالاً در محدوده *192.168.x.y* است. برای بدست آوردن آدرس فوق، فرمان زیر را در کنسول ماشین مجازی وارد کنید:

```
$ /sbin/ifconfig eth0
```

- اگر از *VirtualBox* استفاده می‌کنید و کارت شبکه *eth1* شما در حالت *host-only* قرار دارد، باید از فرمان زیر استفاده کنید:

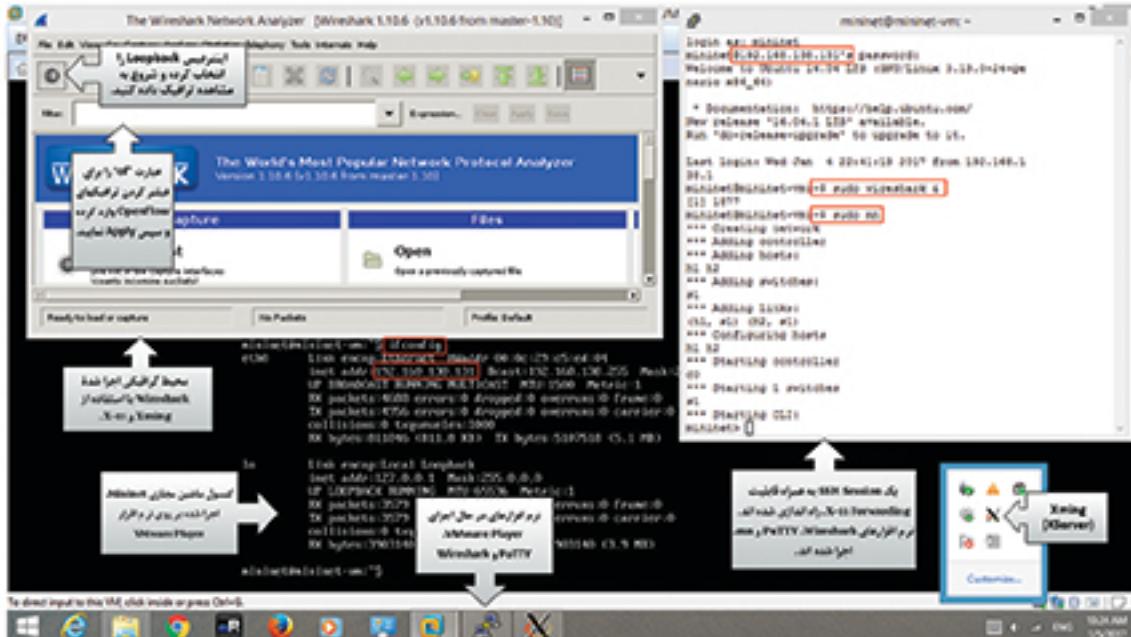
```
$ /sbin/ifconfig eth1 eth1
```

- با این فرض که ماشین مجازی بر روی دسکتاپ خودتان و بصورت محلی اجرا می‌شود و اقدامات احتیاطی اضافی در مورد اتصال *X - ssh* ضروری نیست، می‌توانید با استفاده از فرمان زیر، یک ارتباط SSH با ماشین مجازی ایجاد کنید.

```
ssh -Y mininet@192.168.44.128
```

- فراموش نکنید، شما باید آدرس IP را با آدرسی که به عنوان خروجی فرمان `ifconfig` دریافت کرده‌اید، تعویض کنید. ابزارهایی که ما در این بخش برای راه اندازی Mininet به آنها نیاز داریم، شامل ماشین مجازی Mininet در VMware Player، نرم افزار PuTTY و Xming است. گزینه `X-11 Forwarding` (با گزینه `X-11 Forwarding` فعال شده) و ابزاری برقراری ارتباط SSH (با گزینه `X-11 Forwarding` برای اطلاعات بیشتر به بخش یادداشت بعدی مراجعه کنید) امکان اجرای برنامه‌ها را با خروجی گرافیکی فراهم می‌کند. برای مثال، Wireshark که از قبل در فایل ماشین مجازی Mininet نصب و اضافه شده است، با قابلیت X-11 می‌تواند بصورت گرافیکی اجرا شود. در اسکرین شات زیر می‌توانید محیط آزمایشی مبتنی بر VMware Player، ماشین مجازی Mininet (X-server)، Xming (X-server) و نرم افزار PuTTY را مشاهده کنید.
- در اسکرین شات زیر می‌توانید مشاهده کنید که ما با استفاده از نرم افزار PuTTY وارد ماشین مجازی Mininet شدیم و سپس Wireshark را به عنوان نرم افزاری که در پس زمینه می‌تواند اجرا شود (با استفاده از فرمان `& sudo wireshark`)، راه اندازی کردیم. بدلیل فعل بودن قابلیت `X-11 Forwarding` محیط گرافیکی Wireshark به شکل یک پنجره جداگانه ظاهر می‌شود.

۲-۲
آزمایشگاه
Mininet با



قبل از راه اندازی Mininet، باید به گزینه Capture در Wireshark رفته، اینترفیس Loopback را انتخاب کرده و دریافت ترافیک را آغاز کنید. به منظور نمایش ترافیک مربوط به OpenFlow، باید حروف `of` را که اختصار پروتکل OpenFlow است در قسمت منوی Filter در Wireshark اضافه کنید و روی دکمه Apply کلیک کنید. این کار به Wireshark دستور می‌دهد تا تنها ترافیک مربوط به OpenFlow را نمایش دهد. از آنجاییکه Mininet هنوز شروع به کار نکرده، نباید هیچ بسته OpenFlow در پنجره اصلی Wireshark نمایش داده شود. در بخش بعدی بطور آزمایشی Mininet را اجرا کرده و شبکه‌ایی ساده در آن ایجاد می‌کنیم.

ماشین مجازی Mininet دارای مدیریت دسکتاب نیست. خروجی گرافیکی باید از طریق X11-Forwarding بوسیله ارتباط SSH ارسال شود. شما می‌توانید در آدرس زیر در خصوص پرسشهای متداول برای فعال کردن X-11 Forwarding مکاتبه کنید. نصب صحیح X11 شما را قادر می‌سازد تا دیگر برنامه‌هایی که دارای محیط گرافیکی بوده و همچنین برنامه‌های مدل کننده ترمینال Xterm (که در ادامه این فصل استفاده می‌شوند) را اجرا کنید.



<https://github.com/mininet/mininet/wiki/FAQ#wiki-x11-forwarding>

تجربه‌ای با Mininet

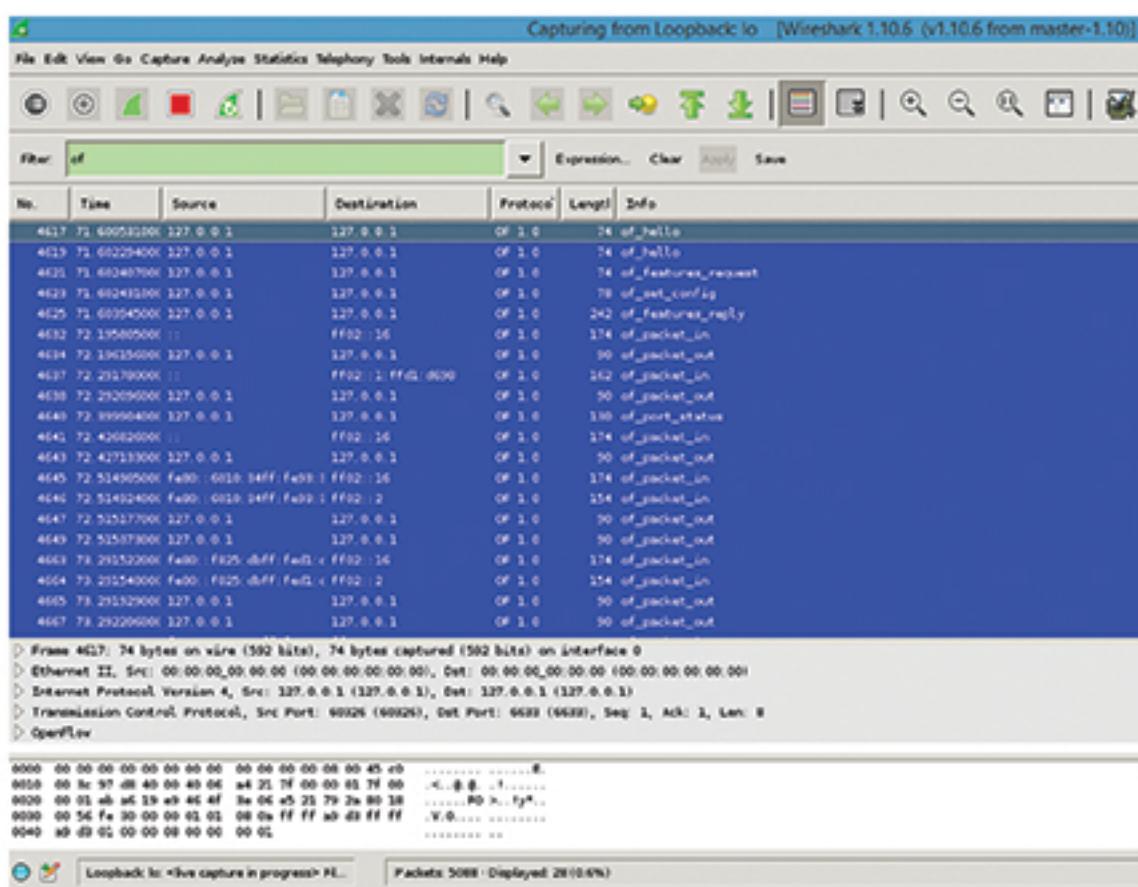
نرم افزار Mininet شما را قادر می‌سازد تا نمونه اولیه OpenFlow را به سرعت ایجاد کرده، آن را سفارشی سازی (Customize) کنید، به تعامل با آن پردازید و در نهایت آن را به اشتراک بگذارید. ابزار خط فرمان Mininet می‌تواند برای ایجاد یک شبکه، شامل تعداد زیادی میزبان و سوئیچ، مورد استفاده قرار گیرد. این خط فرمان به شما اجازه می‌دهد که تمام شبکه مجازی خود را از یک کنسول واحد، کنترل و مدیریت کنید. علاوه بر آن، API در Mininet به شما اجازه می‌دهد اپلیکیشن‌های شبکه سفارشی را با چند خط اسکریپت به زبان Python ایجاد کرده و توسعه دهید. زمانیکه نمونه اولیه بر روی Mininet مورد آزمایش قرار گرفته و نتایج مطلوبی را به همراه داشت، می‌توان آنرا در شبکه واقعی نیز ایجاد کرده و گسترش داد. در این نمونه آزمایش، ما توپولوژی پیش فرض Mininet را با استفاده از خط فرمان `$ sudo mn` آماده خواهیم کرد. این توپولوژی شامل یک سوئیچ OpenFlow متصل به دو میزبان، به همراه یک کنترلر اصلی OpenFlow است. توپولوژی پیش فرض می‌تواند از روش دیگر و با استفاده از خط فرمان `--topo=minimal` ایجاد شود. توپولوژیهای دیگری نیز در Mininet بدون نیاز به برنامه نویسی، آماده استفاده هستند. به قسمت `--topo` در خروجی `mn -h` توجه کنید. شما

می‌توانید گره‌ها (Node)، لینکها و اطلاعات ابتدایی درباره کلیه گره‌ها در تنظیمات مربوطه را، به ترتیب با استفاده از فرمانهای زیر نمایش دهید:

```
mininet> nodes
mininet> net
mininet> dump
```

به محض اجرای توپولوژی پیش فرض در محیط Mininet، کنترلر و سوتیج OpenFlow یک پروتکل Wireshark را آغاز کرده، که می‌توان این عملکرد را در پنجره Capture مربوط به نرم افزار OpenFlow مشاهده و دریافت کرد. اسکرین شات زیر ترافیک دریافت شده را نشان می‌دهد که به موجب آن پیغامهای [Hello](#)، [Feature Request](#)، [Feature Reply](#) و چندین پیغام Packet-in، Packet-out در آن قابل مشاهده است. این وضعیت تائید می‌کند که سوتیج OpenFlow به کنترلر OpenFlow متصل است.

شکل ۲-۳
ترافیک دریافت شده
در
OpenFlow
Wireshark



اگر اولین کلمه تایپ شده در خط فرمان Mininet، نام یک میزبان، سوتیج، یا نام یک کنترلر باشد، فرمان بر اساس آن گره اجرا می‌شود. برای مثال، شما می‌توانید اینترفیس اترنت و اینترفیس Loopback میزبان (h1) را با استفاده از فرمان زیر مشاهده کنید:

```
mininet> h1 ifconfig -a
```

حال می‌توانیم با یک فرمان ساده Ping، اتصال هر کدام از میزبانها را بررسی کنیم:

```
mininet> h1 ping -c 1 h2
```

این فرمان یک تک بسته Ping را از میزبان ۱ (h1) به سمت میزبان ۲ (h2) ارسال می‌کند. ارسال بسته‌های ARP میزبان ۱ (h1) به سمت میزبان ۲ (h2)، باعث می‌شود یک پیغام `packet_in` به کنترلر OpenFlow ارسال شود. سپس کنترلر یک پیغام `packet_out` به سمت سوئیچ ارسال می‌کند تا سوئیچ بسته Broadcast کرده و بر روی همه پورتهای دیگر خود پخش کند. میزبان ۲، بسته ARP را مشاهده کرده و یک Broadcast به سمت سوئیچ ارسال می‌کند. این پاسخ به دست کنترلر می‌رسد، سپس کنترلر آن را به میزبان ۱ می‌فرستد و یک ورودی جریان را به جدول جریان سوئیچ `S1` (سوئیچ OpenFlow) اضافه می‌کند.

222	22.344214000	127.0.0.3	127.0.0.1	0FP	74 Echo Reply (190/180)
285	17.341212000	127.0.0.3	127.0.0.1	0FP	74 Echo Request (190/180)
300	17.343643000	127.0.0.3	127.0.0.1	0FP	74 Echo Reply (190/180)
206	26.550500000	52.15.87.40	127.0.0.1	Broadcast	0FP+MP
206	26.550500000	127.0.0.3	127.0.0.1	0FP	136 Packet In (400) (Buf[0]=2405) (400) wr:Mac:16.0.0.27 Tgt:15.0.0.1
206	26.550500000	127.0.0.3	127.0.0.1	0FP	136 Packet Out (400) (Buf[0]=2405) (400)
299	29.343232000	62.17.73.40.14.25	127.0.0.1	0FP+MP	126 Packet In (400) (Buf[0]=2043) (400) wr:18.0.0.2 1a:42.17.73.40.14.25
230	20.352526000	127.0.0.3	127.0.0.1	0FP	146 Flow Add (100) (100)
230	20.352526000	127.0.0.3	127.0.0.1	0FP+TDP	182 Packet In (400) (Buf[0]=2070) (1000) wr:Echo:Spanning-request: 1a:0x00000000; seq:1/256; vif:64
230	20.352526000	127.0.0.3	127.0.0.1	0FP	146 Flow Del (100) (100)
230	25.342722000	127.0.0.3	127.0.0.1	0FP	74 (TCP ACKed unsegmented) Echo Request (190/180)
230	25.343329000	127.0.0.3	127.0.0.1	0FP	74 (TCP ACKed unsegmented) (TOP Previous request not captured) Echo Reply (190/180)
245	25.557500000	62.17.73.40.14.25	127.0.0.1	0FP+MP	126 Packet In (400) (Buf[0]=2030) (400) wr:Mac:16.0.0.17 Tgt:15.0.0.2
250	25.554216000	127.0.0.3	127.0.0.1	0FP	146 Flow Add (100) (100)
250	25.554433000	52.15.87.40	127.0.0.1	0FP+MP	126 Packet In (400) (Buf[0]=2790) (400) wr:18.0.0.1 1a:52.15.87.40
250	25.554433000	127.0.0.3	127.0.0.1	0FP	146 Flow Del (100) (100)
324	30.343237000	127.0.0.3	127.0.0.1	0FP	74 Echo Request (190/180)
325	30.343239000	127.0.0.3	127.0.0.1	0FP	74 Echo Reply (190/180)

شکل ۲-۴

حال میزبان ۱، آدرس IP میزبان ۲ را دانسته و می‌تواند Ping خود را از طریق یک درخواست Echo Request ارسال کند. این درخواست و درخواست مربوطه از سمت میزبان ۲، هر دو به کنترلر می‌روند و کنترلر، ورودی جریان مربوط به این دو جریان ترافیکی را در جدول جریان سوئیچ قرار می‌دهد. ارسال پسته‌های بعدی بین این دو گره، آغاز می‌شود. در تنظیمات ما، زمان گزارش شده برای فرمان Ping حدود 3.93 ms است. همان فرمان Ping را مجددآ تکرار می‌کنیم:

```
mininet> h1 ping -c 1 h2
```

برای دومین فرمان Ping، زمان به 0.25 ms کاهش می‌یابد. یک ورودی جریان که ترافیک Ping مربوطه را پوشش می‌دهد از قبل در سوئیچ نصب شده است، بنابراین هیچ کنترل ترافیکی‌ای توسط کنترلر انجام نشده و پسته‌ها به سرعت از سوئیچ گذشته و به سمت مقصد ارسال می‌شوند. یک روش

ساده‌تر برای اجرای این آزمایش، استفاده از فرمان `pingall` است که در خط فرمان Mininet تعبیه شده است. در این روش یک Ping بین تمام جفت المانها اجرا می‌شود. آزمایش مفید دیگر تست Regression^{۱۷} بوده که درون Mininet وجود دارد. فرمان زیر یک توپولوژی با حداقل عناصر لازم را ایجاد کرده، کنترلر اصلی OpenFlow را راه اندازی کرده، یک تست Ping بین تمام جفت المانها اجرا کرده و در نهایت توپولوژی و کنترلر را از بین می‌برد.

```
$ sudo mn --test pingpair
```

آزمایش مفید دیگر ارزیابی کارائی بوده که با ابزار^{۱۸} `iperf` انجام می‌شود.

```
$ sudo mn --test iperf
```

این فرمانها برای کامل شدن، به چند ثانیه زمان نیاز دارند. فرمان فوق همان توپولوژی Mininet یعنی یک کنترلر، یک سوئیچ و دو میزبان را ایجاد می‌کند. سپس یک سرور `iperf` را روی اولین میزبان (به عنوان مثال `h1`) اجرا می‌کند، و در ادامه یک کلاینت `iperf` را روی دومین میزبان (به عنوان مثال `h2`) اجرا کرده و پهنای باند ترافیک TCP را بین این دو میزبان محاسبه و گزارش می‌کند.

با استفاده از Python API متعلق به Mininet، تعریف توپولوژیهای متدائل و مرسوم، جهت انجام آزمایشهای گوناگون، امکان پذیر می‌شود. نمونه‌ای از این نوع توپولوژیها در آدرس

`~/mininet/custom/topo-2sw-2host.py` در سرور Mininet موجود بوده و قابل دسترس است. این نمونه توپولوژی، از دو سوئیچ و دو میزبان استفاده می‌کند. بر این اساس دو سوئیچ را مستقیماً به یکدیگر متصل کرده و به هر یک از این سوئیچ‌ها یک میزبان را وصل می‌کند:

```
"""Custom topology example
Two directly connected switches plus a host for each switch:
    host --- switch --- switch --- host
    h1 <-> s3 <-> s4 <-> h2
Adding the 'topos' dict with a key/value pair to generate our
newly defined
topology enables one to pass in '--topo=mytopo' from the command
line.
"""
from mininet.topo import Topo
class MyTopo( Topo ):
    "Simple topology example."
```

^{۱۷} تست Regression، به فرایند تست تغییرات برنامه‌های کامپیوتري گفته شده تا بر اساس آن اطمینان حاصل شود برنامه قدیمی تر با تغییرات جدید می‌تواند سازگار باشد. این نوع تست جزو بخش‌های معمول در فرایند توسعه برنامه‌ها است. (متترجم)

^{۱۸} نرم افزار iPerf، ابزاری برای تست کارائی شبکه است. این نرم افزار بر روی پورت‌های TCP با UDP شروع به ایجاد ترافیک داده کرده و پس از آن، میزان گذردهی شبکه را محاسبه می‌کند. این ابزار بصورت Client-Server کار می‌کند و از طریق پارامترهای مختلف که می‌توان برای آن تنظیم کرد، وضعیت شبکه را از جنبه‌های مختلف مورد ارزیابی قرار می‌دهد. (متترجم)

```

def __init__( self ):
    "Create custom topo."
    # Initialize topology
    Topo.__init__( self )
    # Add hosts and switches
    leftHost = self.addHost( 'h1' )
    rightHost = self.addHost( 'h2' )
    leftSwitch = self.addSwitch( 's3' )
    rightSwitch = self.addSwitch( 's4' )
    # Add links
    self.addLink( leftHost, leftSwitch )
    self.addLink( leftSwitch, rightSwitch )
    self.addLink( rightSwitch, rightHost )
topos = { 'mytopo': ( lambda: MyTopo() ) }

```

اسکریپت Python مذکور را می‌توان به عنوان یک پارامتر خط فرمان به Mininet انتقال داد. زمانیکه یک فایل Mininet فراهم می‌شود، واند توپولوژیهای جدید، انواع سوئیچها و تستها را به خط فرمان اضافه کند. به عنوان مثال، یک تست pingall را می‌توان با استفاده از توپولوژی ذکر شده در ادامه، اجرا کرد:

```
$ sudo mn --custom ~/mininet/custom/topo-2sw-2host.py --topo mytopo --test pingall
```

برای اشکال زدایی‌های پیچیده تر و همچنین دسترسی به کنسول میزبان‌ها، سوئیچ‌ها) یا کنترلر(ها)، می‌توانید Mininet را با پارامتر خط فرمان `-x` (که عبارت است از `sudo mn -x`) شروع به اجرا کنید. Xterm‌ها که بصورت Pop Up ظاهر می‌شوند برای اجرای فرمانهای تعاملی مفید هستند. برای مثال در Xterm مشخص شده، با نام Switch: s1(root)، می‌توانید فرمان زیر را اجرا کنید:

```
# dptcl dump-flows tcp:127.0.0.1:6634
```

از آنجا که جدول جریان سوئیچ S1 خالی است، چیزی بر روی صفحه Xterm چاپ نمی‌شود. حال در Xterm میزبان 1 (یعنی `h1`) می‌توانید میزبان 2 (یعنی `h2`) را با استفاده از فرمان `# ping` Ping، 10.0.0.2 کنید. اگر به سراغ Xterm سوئیچ S1 بروید و جدول جریان را نمونه برداری کنید، باید چندین ورودی جریان را مشاهده کنید. همچنین می‌توانید فرمان `dptcl`^{۱۰} که بصورت پیش فرض در Mininet وجود دارد را بکار ببرید.

^{۱۰} ابزاری درون سیستم عامل لینوکس بوده که بسته‌های ورودی و خروجی که از طریق اینترفیس شبکه وارد سیستم عامل می‌شوند را روی صفحه نمایش، نشان می‌دهد. (متترجم)

این بخش مقدمه‌ای فشرده درباره Mininet بود. در فصلهای بعدی ما از Mininet به عنوان بخشی از فعالیتمان برای آزمایش کنترلرهای OpenFlow و توسعه اپلیکیشن‌های شبکه، استفاده خواهیم کرد. خوانندگان علاقمند، می‌توانند جزئیات بیشتری را در وب سایت Mininet به آدرس www.mininet.org یافت کنند.



خلاصه

پیاده‌سازی مرجع سوئیچ OpenFlow عبارت است از `ofdatapath` که جدول جریان را در فضای کاربر پیاده‌سازی می‌کند، `ofprotocol` برنامه‌ایی که عناصر کانال امن سوئیچ اصلی را پیاده‌سازی می‌کند و `dptcl` که ابزاری برای پیکربندی سوئیچ محسوب می‌شود. سه گونه پیام اصلی در پروتکل OpenFlow وجود دارد (کنترلر - به - سوئیچ، پیغام نامتقارن و پیغام همزمان). علاوه بر سوئیچهای سخت افزاری OpenFlow، نوعی پیاده‌سازی نرم افزاری OpenFlow به شکل سوئیچ نرم افزاری وجود دارد. Mininet یک مدل کننده شبکه است، که مجموعه‌ای از میزبانها، سوئیچ‌ها و لینکها را بر روی یک کرنل لینوکس اجرا می‌کند. در این فصل ما با Mininet آشنا شده و آن را به عنوان یک آزمایشگاه OpenFlow، در یک کامپیوتر بکار بردیم. در فصل بعد، ما به مراجع گزینه‌های دیگر کنترلر OpenFlow/SDN خواهیم رفت.