به نام خدای گلم

سلام دوستان مهربونم

امیدوارم همگی خوب و خوب باشین

لازمه قبل از شروع من توضیحاتی پیرامون ترجمه بهتون بدم

اولا معذرت میخوام ترجمه داره یکم طول میکشه و فعلا قسمت اول را که تا صفحه ۴ است را مطالعه کنید تا ترجمه بعدی خدمتتون ارائه کنم.

دوما،چون این یک متن تخصصی است و دارای اصطلاحات شبکه ای زیادی است که در صورتی که اطلاعاتی در مورد آن اصطلاحات نداشته باشیم، فهم جملات برامون بسیار سخت خواهد شد و به همین دلیل سعی کردم تا جایی که امکان دارد در کنار ترجمه، یکسری مطالب تکمیل کننده ای هم اضافه کنم تا مطالب روان تر شود.

و در بعضی قسمتها لینک منابع انگلیسی که مطالب را از آن استخراج کردم را به صورت هایپرلینک مشخص کردم که می توانید اصل مطلب و توضیحات کاملی تری در مورد آن لینک را مشاهده کنید.

معمولا قسمتهای اضافه شده تکمیلی با رنگ آبی مشخص شده است.

سبز باشین

بررسی: SDN

(شبکه های تعریف شده با نرم افزار) Software-Defined Networking

گذشته، حال، و آینده شبکه های قابل برنامه ریزی

چکیده:

ایده شبکه های قابل برنامه ریزی به تازگی با توجه به ظهور SDN شتاب قابل توجهی گرفته است. SDN، اغلب روی بخش اصلی ایده جدید شبکه (قابل برنامه بودن شبکه ها) تمرکز دارد.

SDN وعده داده است که به صورت چشمگیری مدیریت شبکه را آسان کند و همچنین با استفاده از قابلیت برنامه نویسی شبکه، امکان پیاده سازی ایده های جدید در شبکه را بوجود آورد.

این مقاله به بررسی آخرین پیشرفتها در زمینه SDN می پردازد.

در این مقاله سیر مراحل ایده های اولیه تا پیشرفتهای اخیر را بررسی می کنیم و سپس ساختار SDN را توضیح می دهیم و همچنین به طور ویژه در مورد استاندارد Openflow صحبت می کنیم.

بحث در مورد جایگزینی های فعلی در شبکه برای پیاده سازی و تست سرویس ها و پروتکل های مبتنی بر SDN است و همچنین APP های فعلی (منظور از app برنامه های کاربردی است که ما می نویسیم مثل mac-learning یا فایروال و...) و آینده SDN را بررسی می کنیم.

کلمات کلیدی:

SDN، شبکه های قابل برنامه ریزی،بررسی،

Data Plane (قسمتی از شبکه که ترافیک کاربر را جابجا می کند)،

Control Plane (قسمتی از شبکه است که ترافیک های کاربر را کنترل و مدیریت می کند)

Virtualization (مجازی سازی)

مقدمه:

شبکه های کامپیوتری به طور معمول از تعداد زیادی از دستگاه های شبکه مانند روترها، سوئیچ ها و انواع متعددی از middleboxes ساخته می شود و همچنین پروتکل های پیچیده و مختلفی بر روی آن اجرا می شود.

Middleboxes چیست: در دنیای شبکه به تجهیزاتی گفته می شود که برای رسیدن به اهداف خاص در ترافیک ها دستکاری می کند و همچنین ترافیک را بازرسی و یا انتقال میدهد، مانند فایروال که مثل ایست و بازرسی پلیس عمل میکند.

اپراتورهای شبکه باید با استفاده شرایط شبکه با پیکربندی ها و سیاست هایی که در نظر می گیرد اتفاقات شبکه را کنترل و برنامه ریزی کند در واقع اپراتور شبکه مسئول پیاده سازی این سیاستها و پیکربندی هاست و او باید به صورت دستی سیاستهای سازمان را به دستورات پیکربندی تبدیل کند.

اغلب اپراتورهای برای انجام چنین کاریی به ابزارهایی زیادی دسترسی ندارند و معمولا محدود هستند، به عنوان مثال زمانی که شما سوئیچ سیسکو دارید و می خواهید تنظیماتی روی آن انجام دهید، فقط می توانید تنظیماتی انجام دهید که قبلا شرکت سیسکو در اختیار شما قرار داده، پس شما محدود هستین به امکاناتی که سیسکو در اختیار شما قرار داده پس شما محدود هستید.

پس نتیجه می گیریم:

مدیریت و پیکربندی شبکه برای مدیران و اپراتورهای شبکه یک چالش بزرگی است و امکان خطا در شبکه بسیار است.

یکی دیگر از چالش های بزرگ که محققان و دکترین شبکه با آن مواجه شدند و تقریبا قابل برطرف شدن هم نیست و به عنوان یک سنگ بزرگ در اینترنت اشاره می شود و آن موضوع زیرساخت عظیم شبکه اینترنت است.
چون این زیرساخت شامل سیر عظیمی از تجهیزات و پروتکل های مختلف شبکه است، همیشه توسعه و تغییرات در آنها یک کار بسیار بسیار دشواری است.
هرچند برنامه ها و سرویس های مختلفی هر روز در حال اضافه شدن به اینترنت است و همچنین به طور فزاینده ای خدمات و پیچیده تر و سخت تر می شود ولی واقعا ضروری است که شبکه اینترنت قابلیت توسعه با توجه به چالشهای مختلف را داشته باشد.
پس ما باید به سمتی برویم که بتوانیم در سریع ترین زمان و با بالاترین کیفیت و بهروری نسبت به چالشهای که پیش خواهد آمد، بهترین برخورد را داشته باشیم.

ایده شبکه های قابل برنامه ریزی، به عنوان راه حلی برای توسعه آسان تر شبکه ارائه شد. به طور خاص شبکه های نرم افزار محور (SDN)، یک الگویی از شبکه های جدید است که در آن وظیفه سخت افزار فقط حمل و نقل است و دیگر وظیفه تصمیم گیری ندارد و این وظیفه تصمیم گیری از سخت افزار جدا می شود(این وظیفه به عهده کنترللر داده می شود مثل OpenDayLight که در پروژه در مورد آن بحث کردیم).
این وعده ای که SDN به ما داده به طرز چشمگیری مدیریت شبکه را ساده می کند و همچنین این امکان فراهم می شود که ما بتونیم ایده های خودمان را به راحتی در شبکه پیاده سازی کنیم.
ایده اصلی این است که از این به بعد، به توسعه دهندگان نرم افزار این اجازه داده می شود که با تکیه بر منابع سخت افزاری شبکه، بر روی داده های عبوری مدیریت کنند و بتوانند با برنامه نویسی، سیاستهای خود را در شبکه پیاده سازی کنند.
در SDN، مغز متفکر شبکه کنترللر است (سطح کنترل Control Plane) و سخت افزار وظیفه حمل و نقل ساده بسته ها را دارد (سطح داده Data Plane) که همین بسته توسط کنترللر کنترل می شوند.
SDN در حال حاضر مورد توجه دانشگاه و صنعت می باشد. به تازگی گروهی از اپراتورهای شبکه،توسعه دهندگان، و فروشندگان تجهیزات شبکه یک موسسه غیرانتفاعی با عنوان Open Network Foundation(ONF) ایجاد کرده اند و هدف آنها استانداردسازی و بهبود شبکه SDN و پروتکل Openflow ئ تکنولوژی های مرتبط با آن بوده است که این موسسه توسط شرکتهای زیر تاسیس شده است:
Deutsche Telekom,Facebook, Google, Microsoft, Verizon, and Yahoo

بحث SDN هنوز کاملا جدید است و با سرعت بسیار زیادی در حال رشد است. با این حال هنوز چالشهای پژوهشی مهمی برای مخاطبین آن وجود دارد.
ما در این مقاله با توجه پیشنه های تاریخی به بررسی آخرین پیشرفتها در زمینه شبکه های قابل برنامه ریزی می پردازیم و همچنین جزئیات معماری SDN را تشریح می کنیم. ساختار مقاله به صورت زیر است:

در بخش دوم:
شروع بحث ما، تمرکز بر روی شبکه های قابل برنامه ریزی است.

در بخش سوم:
مرور کلی بر SDN و معماری آن و همچنین در مورد پروتکل OpenFlow هم بحث می کنیم. (ترجمه اولیه تا آخر همین بخش می باشد)

در بخش چهارم:

ما در مورد پلتفرم های موجود (بسترهای موجود) برای توسعه و تست SDN بحث می کنیم که شامل: ابزارهای شبیه سازی(Mininet)، پیاده سازی کنترللرهای SDN (به عنوان مثال OpenDayLight) و ابزارهای بررسی و خطایابی (مثل WireShark)

بخش پنجم:

در این بخش بر روی برنامه های مختلف SDN در محیطهایی همچون دیتاسنتر ها و شبکه های بی سیم بحث می کنیم.
منظور از برنامه های مختلف SDN همان App هایی است که ما قرار است در پروژه بنویسیم (مثلا Mac Learning)

بخش ششم:

در فصل آخر بر روی چالش های پژوهشی و مسیر آینده SDN بحث می کنیم.

بخش دوم:

شبکه های قابل برنامه ریزی

SDN دارای پتانسیل بسیار زیادی برای تغییر مسیر شبکه دارد و پروتکل OpenFlow به طور ویژه به عنوان ریشه این ایده جدید در شبکه می باشد. مزایای این ایده جدید چیست:

- کنترل مرکزی شبکه
- الگوریتم های ساده
- صرفه جویی و بهینه شدن تجهیزات شبکه
- حذف دستگاههای میانی (Middleboxes) مثلا دیگر نیاز به دستگاه فایروال نداریم چون کنترل مرکزی توسط کنترلرر انجام می شود.
- امکان طراحی و توسعه برنامه های Third-party

Third-party چیست:

منظور از این عبارت نرم‌افزارهایی هست که توسط یک شرکت سوم بر روی یک بستر منتشر می‌شوند.
برای مثال نرم‌افزارهای موجود در فیس‌بوک یا گوگل پلاس (بازی‌ها و ...) نرم‌افزارهای Third Party هستند؛ چون توسط خود این شرکت‌ها یعنی فیس‌بوک یا گوگل پلاس طراحی نشده‌اند توسط شرکتهای دیگر ساخته می شود ولی با برند مثلا گوگل.
در بحث SDN، هم وقتی می گوییم برنامه های Third-party یعنی برنامه هایی که توسط خود SDN نوشته نشده است و ممکن است شرکتهای دیگر مثل سیسکو،اینتل و یا خود شما نوشته شود (مثل ما که قرار است در پروژه یک فایروال کوچک بنویسیم. پس ما هم داریم یک APP یا همان برنامه های Third-party می نویسیم.)

در حالیکه الان OpenFlow توجه صنعت را به خود جلب کرده ما باید بدانیم که این دو ایده برنامه ریزی شبکه ها و کنترل متمرکز آنها از چندیدن سال قبل بوده است، در این بخش ما تلاش می کنیم تا شما را با کلیات شبکه های قابل برنامه ریزی و همچنین اجزای تشکیل دهنده مدل فعلی SDN را از ابتدا تا بحال آشنا کنیم که این موارد که ذکر می کنیم، پایه و اساس بسیاری از ایده های هستند که ما امروزه مشاهده می کنیم.

مواردی که در این بخش توضیح خواهیم داد:

a) Open Signaling
b) Active Networking
c) DCAN
d) 4D Project
e) NETCONF
f) Ethane

A) Open Signaling سیگنالینگ باز

کارگروه OPENSIG در سال 1995 با راه اندازی یکسری کارگاههای اختصاصی شروع به فعالیتهای زیر کرد:

- ساخت دستگاههای خودپرداز(ATM)
- اینترنت و شبکه های تلفن همراه خیلی باز، توسعه پذیر و قابل برنامه ریزی

آنها معتقد بودند که سخت افزارها و کنترل نرم افزاری نیاز است که از هم جدا باشند. اما به چالش کشیدن این بحث برای تحقق آن بود.

Vertical integration: یک عملیاتی است که می گوید پیکربندی و سیاست گذاری چندین سوئیچ و روتر توسط یک کنترللر کننده مرکزی مدیریت شود (مانند OpenDayLight) و هر سوئیچ و روتر جدا جدا تنظیم نمی شوند.

به این دلیل است که عمدتا سوئیچ ها و روترهای یکپارچه که ماهیتا بسته ساخته شده اند(یعنی محدود هستند و ما نمی توانیم در سیستم نرم افزاری آنها تغییرات دهیم)، پیاده سازی سریع سرویس ها در این نوع شبکه غیر ممکن است.

در کل هسته پیشنهادی آنها، دسترسی باز و بدون محدودیت به سخت افزارهای شبکه است. (یعنی این امکان فراهم شود که اپراتورهای شبکه بتوانند براحتی در شبکه تغییرات خود را با برنامه نویسی اعمال کنند)

انگیزه های بوجود آمده برای این ایده باعث شد تا کارگروه IETF ایجاد شود که این کارگروه موفق به ساخت یک پروتکل به نام GSMP(General Switch Management Protocol) شد.

که این پروتکل امکانات زیر را در شبکه فراهم کرد:

- اضافه و حذف کردن زیرشاخه های ارتباطات Multicast (چندبخشی)
- مدیریت پورت های سوئیچ
- درخواست اطلاعات تنظیمات انجام شده روی سوئیچ
- درخواست و حذف رزرو منابع سوئیچ (مثلا با برنامه نویسی ما یکسری منابع سوئیچ را رزرو می کنیم)
- آمار گیری از وضعیت سوئیچ ها

این کارگروه بعد از تلاشهای مختلف آخرین پیشنهاد خودشون را تحت استاندارد GSMPv3 در ژوئن 2002 منتشر کرند.

B) Active Networking شبکه های فعال

در اواسط دهه 1990، Active Networking ایده ای را بری شبکه های زیرساخت داد که به واسطه آن امکان برنامه ریزی و سفارشی سرویس ها فراهم شد.

تعریف در سایت ویکی پدیا: یک الگوی ارتباطی است که به بسته ها اجازه می دهد در سرتاسر شبکه جریان داشته باشند و بصورت پویا و اتوماتیک تغییرات را در شبکه اعمال کند.

دو روش اصلی برای این کار وجود دارد:

۱- سوئیچ های قابل برنامه ریزی توسط کاربر:

برای ارسال داده ها از یک کانال و برای مدیریت از یک کانال اختصاصی مجزا استفاده می شود.

In-band= زمانی استفاده می شود که بخواهیم هم داده و هم جریان کنترلی روی یک مسیر باشند

Out-of-band= زمانی استفاده می شود که بخوهیم جریان داده و کنترل از هم جدا باشند.

۲- کپسول

یک کپسول شامل اطلاعات کاربر و کدهای اجرایی است.
این کپسول بعد از ساخته شدن بر روی شبکه ارسال می شود و به همه نودهای فعال تحویل داده می شود، بعد از اینکه نود فعال کپسول را دریافت کرد آن را باز کرده و تفسیر می کند و بعد کد مربوطه را اجرا می کند.

با توجه به فعالیتهای قابل توجهی که در این زمینه انجام شد، ولی Active Networking هرگز به جمع بندی کامل نرسید تا در شبکه ها و صنعت به کار گرفته شود ، که عمدتا به دلیل نگرانی های امنیتی و کارایی آن بود.

DCAN (C

ابتکار دیگری که در اواسط 1990 صورت گرفت، تحول کنترل شبکه های ATM بود با عنوان
DCAN (Devolved Control of ATM Networks)
هدف اصلی این پروژه طراحی و توسعه زیرساختهای لازم برای کنترل مقیاس پذیری و مدیریت شبکه های ATM بود.
با این فرض که؛ کنترل و مدیریت دستگاهها، باید از خود دستگاهها جدا شود و به دستگاهها یا موجودیتهای خارجی دیگری واگذار شود. که اساسا مفهوم پشت بحث SDN همین است.
DCAN یک پروتکل ساده بین مدیر و شبکه است.
در جواب اینکه، امروزه در طرح های پیشنهای مثل Openflow چه اتفاقاتی می افتد؟ بسیاری از جواب های خود را می توانید در پروژه های DCAN دنبال کنید.

4D Project (C

پروژه 4 بعدی در سال 2004 شروع به کار کرد.
این طرح با توجه به قابلیتهای گذشته و براساس اصول اساسی جدید دوباره طراحی شد که او نیز بر جدایی بین تصمیم گیری و پروتکل های حاکم بر شبکه تاکید دارد.
او یک نگاه کلی تصمیم گیری بر روی شبکه را پیشنهاد داد.
هدف اصلی 4D Project کنترل حمل و نقل در شبکه زیرساخت بود، که این ایده تاثیر مستقیمی بر آثار بعدی مثل NOX داشت.

این پروژه دارای چهار لایه بود:
Data Plane: پردازش بسته ها بر اساس قوانین پیکربندی
Discovery Plane:جمع آوری آمارهای ترافیکی و توپولوژی شبکه
Dissemination Plane: وظیفه نصب قوانینی که باید بعدا پردازش شود.
Decision Plane: شامل کنترل منطقی مرکزی است.

مفوم واژه طرح Clean-Slate: یک سیستم برای ارائه طرح بصورت انتزاعی یا بهبود عملکرد خود، با توجه به قابلیتهای مشابه قبلی و براساس اصول اساسی جدید دوباره طراحی میشود.

NETCONF (D

به عنوان یک پروتکل مدیریت و اصلاح تنظیمات تجهیزات شبکه در سال 2006 ارائه شد.
این پروتکل به تجهیزات شبکه این امکان را می دهد از طریق یک API ، اطلاعات پیکربندی را ارسال و دریافت کند.
یکی دیگر از پروتکل های مدیریت، که از گذشته تابحال بصورت گسترده توسعه داشته، SNMP است.
SNMP در اواخر دهه ۸۰ مطرح شد و نشان داد که یکی از پروتکل های مدیریتی محبوب شبکه است.

که با استفاده از رابط SMI داده‌های موجود در MIB را واکشی می کند.

SMI(Structured Management Interface)= یک رابط مدیریتی ساختاریافته است.
MIB(Management Information Base)= شامل اطلاعات پایه‌ای مدیریتی است.

که آن می‌تواند برای تغییر مقادیر MIB به منظور اصلاح تنظیمات مدیریتی به خوبی مورد استفاده قرار بگیرد.
بعدها مشخص شد، علرغم آنچه که در اصل برای SNMP در نظر گرفته شده بود،از SNMP برای برای پیکربندی تجهیزات شبکه استفاده نمی‌شود و از این ابزار بیشتر به عنوان استراق سمع در شبکه استفاده می شود، علاوه بر کاستی ها متعددی که این پروتکل دارد، مهمترین مشکل آن عدم امنیت کافی است.

در آن زمان NETCONF توسط مؤسسه IETF پیشنهاد شد،اکثرا NETCONF را به عنوان یک رویکرد جدید در مدیریت شبکه برای رفع کاستی های SNMP در شبکه تلقی می کردند.
گرچه NETCONF، با هدف ساده سازی انجام پیکربندی های دوباره، تغییرات را به عنوان Building-Block (اجزای سازنده ساده) برای مدیریت اعمال می‌کرد ولی جدایی بین داده‌ها و سطح کنترل وجود نداردکه در مورد SNMP هم دقیقاً همینطور بود.
پس SNMP و NETCONF داده‌ها و سطح کنترل را از هم جدا نمی‌کنند ولی یک ابزار مدیریتی برای انجام تغییرات مدیریتی در شبکه هستند.

مفهوم Building Block:شبکه از تعداد کمی کامپوننت استاندارد که ساده تر و ارزان تر از خیلی کامپوننت های دیگر در شبکه ها هستند استفاده می کند.

یک شبکه ای که با NETCONF ایجاد شده است نباید به عنوان یک شبکه برنامه ریزی کامل تلقی شود.
طراحی آن در درجه اول برای کمک به پیکربندی خودکار است نه برای کنترل مستقیم تجهیزات شبکه، که آن قدرت پیاده‌سازی سریع و قابلیت برنامه نویسی نوآورانه برای سرویس ها و برنامه‌های کاربردی را ندارد.
به طور ساده می‌توان گفت: خود NETCONF ابزای نیست که بشود با آن برنامه برای شبکه نوشت فقط او یکسری تنظیماتی که ما بهش می‌دهیم را روی شبکه به صورت اتوماتیک اعمال می کند.

در کل می توان گفت NETCONF و SNMP ابزارهای مدیریتی مفید هستند که ممکن است هردو به صورت موازی در شبکه استفاده شوند و این دو پشتیبان شبکه های قابل برنامه ریزی هم هستند.

کارگروه NETCONF در حال حاضر هم فعال هستند و آخرین نسخه ای که ارایه کردند هم مربوط به ژوئن 2011 می باشد.

F) Ethan اتان

پایه اصلی پروتکل OpenFlow یا به عبارتی SDN، پروژه SANE/Ethan بود.
هدف Ethan، تمرکز بر روی یک کنترلر متمرکز برای مدیریت سیاستها و امنیت در یک شبکه بود.
Ethan به مانند SDN، به دوقسمت کاری تقسیم میشد:

۱- یک کنترللر برای تصمیم گیری

۲- سوئیچ Ethan که تشکیل شده است از یک جدول Flowو یک کانال امن برای کنترللر

❖ معماری SDN (شبکه تعریف شده با نرم افزار)

مفهوم Host در شبکه:

هاست در شبکه به یک کامپیوتر یا یک دستگاه دیگر گفته می شود که به شبکه ما وصل است. مثلا: لپتاپ،تبلت،موبایل،پرینتر تحت شبکه، دستگاه حضور و غیاب تحت شبکه اینها همه هاست هستند منظور تمام تجهیزاتی هستند که به شبکه متصل شدند.

شبکه های ارتباطی داده به معمولا از دستگاه کاربر نهایی، یا هاست های متصل شده به زیرساخت های شبکه می باشد.
این زیرساخت شامل:

- هاست ها
- اجزای سوئیچینگ مانند روترها و سوئیچ ها
- لینکهای ارتباطی که وظیفه حمل داده ها بین هاست ها را دارند.

معمولا روترها و سوئیچ ها Closed (منظور،بسته و یا محدود هستند) هستند،اغلب توسط شرکتهای سازنده محدود شده اند و شما در سیستم عامل آنها نمی توانید دخل و تصرفی داشته باشید و فقط مجبورید از امکاناتی که آنها در اختیار شما قرار دادند استفاده کنید به همین دلیل است که می گویند سوئیچ های فعلی شبکه "بسته" هستند.
بنابراین زمانی که یکبار این سوئیچ و روترهای "بسته" در شبکه مستقر و بکار گرفته شوند، توسعه شبکه با توجه به زیرساخت فعلی بسیار سخت است چون این سوئیچ ها محدود هستند دست ما را هم در پیکربندی شبکه می بندند.

به عبارت دیگر، زمانی که ما بخواهیم در شبکه، بعضی پروتکل ها را به نسخه های جدید توسعه بدهیم (مثلا IPv6) و یا پروتکل یا سرویس جدیدی را در شبکه پیاده سازی کنیم، این تجهیزات یک مانع غیرقابل عبور در شبکه های فعلی هستند.
اینترنت هم که شبکه ای از شبکه ها است، از این قاعده مستثنی نیست.

مفهوم Ossification:

واقعا بعضی اصطلاحات در شبکه و کلا انگلیسی نمیشه معنی کرد، من خیلی دنبال معادل این کلمه در فارسی بودم ولی چیزی که کامل مفهوم این کلمه را برسونه را پیدا نکردم ولی یه توضیحاتی پیرامون این اصطلاح میدم.
ما در این مقاله به یک اصطلاح اینچنینی می رسیم

Ossification of the Internet

که این یکی از مشکلات بسیار مهم در اینترنت و شبکه های فعلی است که به دلیل وجود همین مشکل متخصصین به فکر SDN افتادند.
شاید بتوان معنی این اصطلاح را اینطور بیان کرد: شبکه اینترنت زُمُخت!!!!
یعنی شبکه اینترنتی که خیلی سخت میشود آن را توسعه داد و نمیشود هر ایده ای را در آن پیدا کرد و اصلا قابل انعطاف پذیر نیست و کلا تو راه بیا نیستتتتتتتتتتت
ولی چرا؟
دلیلش کاملا مشخصه!!!!! بدلیل سوئچ و روترهایی که "بسته" هستند و قابلیت برنامه ریزی ندارند و قابل انعطاف نیستند.
به عبارتی در این شبکه زُمُخت و بد اخلاق ما نمی توانیم به روترها و سوئیچ ها هر دستوری بدیم تا اطلاعت کنند، آنها به طور انحصاری دارند، پادشاهی می کنند و ما اینجا هستیم، آنها را با استفاده از SDN به زیر بکشانیم و جان تازه ای به شبکه ها بدیم ☺

یکی از روش های دور زدن Ossification، استفاده از تجهیزات میانی است (مانند:فایروال،IDS،NAT و...) بالای زیرساخت شبکه قرار می گیرد. که این تجهیزات بعضی دستورات و برنامه های ما را در شبکه اجرا می کنند.
Content Delivery Networks (CDNs) یک نمونه خوبی است که برای اطلاعات بیشتر می توانید در اینترنت جستجو کنید.

SDN به منظور پیاده‌سازی راحت ایده‌های جدید در شبکه و همچنین کنترل و مدیریت ساده بسته های شبکه ارایه شد.
همانطور که در شکل ۱ می بینید، جداسازی بین تجهیزات حمل و نقل داده‌ها و بخش کنترل این امکان را فراهم کرده تا بتوان پروتکل های جدید و برنامه‌های کاربردی براحتی گسترش یابند.
در SDN، درک ساختار شبکه و مدیریت آن آسان شده است و همچنین، وابستگی به سخت افزار را کاهش داده و قابلیت های نرم افزاری و هوشمندی شبکه را با توجه به وجود کنترللر افزایش می دهد.

❖ معماری فعلی SDN

ما در این بخش به بررسی دو معماری شناخته شده SDN می پردازیم.

۱- ForCES
۲- OpenFlow

هر دو این معماری اصل هدف SDN، یعنی جدایی بین سطح کنترل و داده ها را دنبال می کنند. هر دو استاندارد فوق برای تبادل اطلاعات بین دو سطح کنترل و داده هستند. با این حال، آنها از لحاظ فنی در طراحی، معماری، مدل حمل و نقل، و رابط پروتکل بسیار متفاوت هستند که ما در بخش های بعدی این دو معماری را مورد بررسی قرار می دهیم.

- ForCES

این روش توسط کارگروه IETF پیشنهاد شد و این کلمه مخفف: ForCES (**For**warding and **C**ontrol **E**lement **S**eparation)
این کارگروه می گوید که در معماری داخلی تجهیزات شبکه، باید واحد کنترل مرکزی از واحد حمل و نقل داده ها جدا باشد و این دقیقا همان ایده اصلی SDN است که قبلا بارها تکرار کردیم.
با این حال، دستگاههای شبکه هنوز به عنوان یک واحد مستقل است. ایده ای که این کارگروه داشت، ترکیب دستگاه حمل و نقل جدید با کنترل های third-party (کنترل های شخص سوم) در یک دستگاه شبکه است.
بنابراین، واحد کنترل و داده در کنار هم نگه داشته می شوند (به عنوان مثال: در یک اتاق یا یک باکس). ولی در مقابل، در معماری OpenFlow، سطح کنترل به طور کامل از سخت افزارهای شبکه جدا هستند. پس دقت داشته باشید یکی از تفاوتهای این دو معماری در همین مورد است.

ForCES دو واحد منطقی به نام های Forwarding Element (FE) و Control Element (CE) را تعریف می کندکه هر دو برای پیاده سازی پروتکل ارتباطی ForCES است.

- Forwarding Element (FE) : منظور عناصر حمل و نقل بسته ها است

مسئول استفاده از سخت افزارها برای هندل کردن هر بسته است (این هندل یا هدایت در سطح سخت افزار است)

- Control Element (CE) : منظور عناصر کنترلی بسته ها است.
  - کنترل اجرا
  - کنترل توابع سیگنالینگ
  - به FE ها یاد میدهد که چطور بسته ها را جابجا کند.

این پروتکل بر مبنای Master/Slave کار میکند . که در اینجا FE ها به عنوان Slave و CE ها به عنوان Master هستند.
Master: کسی که فرمان می دهد (ارباب)
Slave: کسی که فرمان را اجرا می کند (برده)

یک بخش سازنده بسیار مهم در معماری ForCES، LFB است که مخفف (Logical Function Block) بخش توابع منطقی
تعریف LFB:
بلوک تابعی است که روی FE ها قرار میگیرد، که این بلوک توسط CE ها کنترل می شود.
LFB این امکان را فعال می کند که CE ها بتوانند پیکربندی FE ها را کنترل کند و این که بگوید FE ها چطور بسته ها را پردازش کند.

ForCES از سال ۲۰۰۳ تحت استاندارد قرار گرفت و گروه کاری آنها اسناد مختلفی را انتشار دادند از جمله: تشریح کاربردهای این پروتکل، یک زبان مدل سازی که توابع منطقی را در FE ها تعریف می کند، پروتکلی برای ارتباط بین عناصر Forwarding(حمل و نقل) و کنترل. این کارگروه همچنان فعال است.

❖ OpenFlow

از اصل قاعده SDN، که جداسازی کنترل از حمل و نقل داده هاست پیروی می کند.OpenFlow، شبیه ForCES، نحوه تبادل اطلاعات بین دو Plane را استانداردسازی میکند.
منظور از دو Plane: دو واحد Control Plane و Forwarding Plane است.
در معماری OpenFlow، که در شکل ۲ نشان داده شده است، دستگاه حمل و نقل یا سوئیچ OpenFlow، شامل یک یا چند جدول جریان(Flow Table) و یک لایه انتزاعی است که برای ارتباط ایمن این دستگاه با کنترللر از پروتکل OpenFlow استفاده شده است.

جداول جریان شامل مقادیر جریان هستند، که هرکدام از این مقادیر تعیین می کند که چطور بسته های متعلق به یک جریان (Flow) باید پردازش و ارسال شوند.

عناصر هر Flow معمولا شامل:

۱- **فیلدهای تطابق:**
برای تطابق بسته های ورودی استفاده می شود، فیلدهای تطابق ممکن است شامل اطلاعات پیدا شده در هدر بسته، پورت ورودی و یا Metadata ها باشد.
Metadata (فراداده): به داده هایی گفته می شود که جزئیات یک داده دیگر را تشریح می کند یا به عبارت دیگر به داده هایی گفته می شود که درباره داده های دیگر توضیح می دهد.
در پروژه ما در قسمت Flow چندین فیلد داشتیم که منظور همان است مثلا: آدرس مبداء، آدرس مقصد، نوع پروتکل و ...

۲- **شمارنده ها:**
جمع آوری آمار برای جریان (Flow) خاص، آمارهایی همچون:
- تعداد بسته های دریافتی
- تعداد بایت و مدت زمان جریان (Flow)

۳- **مجموعه ای از دستورالعمل و اقدامات (Actions)**
- زمانی که یک بسته ورودی با یک Flow تعریف شده در جدول جریان انطباق پیدا کرد در این حالت Action مربوطه اعمال می شود.
- مانند پروژه که ما چند Flow در کارگاه ایجاد کردیم (مثلا Drop)

به محض ورود بسته به سوئیچ OpenFlow، فیلدهای هدر بسته دریافت می شوند و با فیلدهای موجود در جدول Flow های ورودی مطابقت داده می شود، اگر یک ورودی تطابق پیدا کرد، سوئیچ یکسری دستورات و اکشن ها را روی آن اعمال می کند.
اگر هم ورودی ها با جدولFlow تطابقت نداشت خود سوئیچ در برابر این ورودی عکس العمل نشان میدهد که آن بستگی دارد که دستورات از قبل تعریف شده برای جدول Table-miss. هر جدول Flow باید شامل یک جدول Miss یا به عبارتی جدول عدم تطابق هم باشد.
که در این جدول برای ورودی ها خاص که تطبیق داده نشده اند یکسری قوانین و عکس العمل ها تعریف شده است مثلا:

- حذف کردن بسته ورودی
- ادامه عمل تطابق روی جدول Flow بعدی
- ارسال بسته به کنترللر با استفاده از پروتکل OpenFlow

البته شایان ذکر است که از نسخه ۱,۱ پروتکل OpenFlow، سیستم چند جدولی و پردازش خط لوله ای را پشتیبانی می کند.
پردازش خط لوله ای:

- تکنیکی است برای تجزیه‌ی یک فرآیند ترتیبی به تعدادی زیرعمل.
- هر زیرفرآیند در یک قطعه‌ی اختصاصی ویژه اجرا می‌شود.

یکی دیگر از امکانات، در مورد سوئیچ های ترکیبی است، به عنوان مثال سوئیچ های که هر دو پورت OpenFlow و غیر OpenFlow را دارند، که بسته هایی که تطابق ندارند را با استفاده از طرح منظم حمل و نقل IP، ارسال می کند.
ارتباط بین کنترللر و سوئیچ بوسیله پروتکل OpenFlow انجام می شود، که مجموعه ای از پیامهایی را تعریف می کند که می تواند بین این موجودیتها (منظور سوئیچ و کنترللر است) روی یک کانال امن جابجا شوند.
شما می توانید با استفاده از پروتکل OpenFlow از راه دور شبکه خود را کنترل کنید، برای مثال: می توانید مقادیر داخل جداول سوئیچ های را حذف و اضافه و آپدیت کنید. که این امر می تواند به صورت دو صورت انجام شود:

۱- واکنشی (یعنی در پاسخ به ورود یک بسته عملی انجام شود)
۲- صورت فعالانه یا اتوماتیک انجام شود

مناظره بین دو پروتکل ForCES و OpenFlow

در این قسمت قصد داریم در مورد شباهتها و تفاوتهای این دو پروتکل بحث کنیم.
در میان تفاوتهایی که این دو دارند، این واقعیت کاملا روشن است که مدل حمل و نقل استفاده شده توسط ForCES متکی بر بلوک تابع منطقی (LFB) ها است، در حالی که OpenFlow از جداول Flow استفاده می کند.
در کل اقدامات و عملیات OpenFlow، در رابطه با یک Flow (جریان) بوده که می تواند کارهای زیر را انجام دهد:

- فراهم شدن امکان کنترل بیشتر
- انعطاف پذیری برای اهداف مدیریتی شبکه (منظور پیاده سازی ایده های جدید است)
- توسعه شبکه

در ForCES، ترکیبی از LFB های مختلف می توانند برای رسیدن به یک هدف مشابه استفاده شوند.
ما دوباره تکرار می کنیم که ForCES از زیربنای مشابه مدل SDN پیروی نمی کند، اما آن میتواند برای رسیدن به اهداف مشابه و پیاده سازی قابلیت های مشابه مورد استفاده قرار بگیرد.

B. Forwarding Devices تجهیزات حمل و نقل