دولت جمهوری اسلامی افغانستان
ادارۀ تعلیمات تخنیکی و مسلکی
معاونیت امور اکادمیک
ریاست نصاب و تربیه معلم

مدیریت سیستم ۲

رشته: کمپیوتر ساینس - دیپارتمنت: شبکه

صنف ۱۴ - سمستر دوم

سال: ۱۳۹۹ هجری شمسی

شناسنامه کتاب

نام کتاب: مدیریت سیستم -۲

رشته: کمپیوتر ساینس

تدوین کننده: پوهنمل یحیی اخلاقی

همکار تدوین کننده: شگوفه حسنی

کمیته نظارت:

- ندیمه سحر رئیس ادارۀ تعلیمات تخنیکی و مسلکی
- عبدالحمید اکبر معاون امور اکادمیک ادارۀ تعلیمات تخنیکی و مسلکی
- حبیب الله فلاح رئیس نصاب و تربیه معلم
- عبدالمتین شریفی آمر انکشاف نصاب تعلیمی، ریاست نصاب و تربیه معلم
- روح الله هوتک آمر طبع و نشر کتب درسی، ریاست نصاب و تربیه معلم
- احمد بشیر هیله‌من مسؤل انکشاف نصاب، پروژه انکشاف مهارت‌های افغانستان
- محمد زمان پویا کارشناس انکشاف نصاب، پروژه انکشاف مهارت‌های افغانستان
- علی خیبر یعقوبی سرپرست مدیریت عمومی تألیف کتب درسی، ریاست نصاب و تربیه معلم

کمیته تصحیح:

- سحر احمدی
- دوکتور سید عارف عارف
- محمد امان هوشمند مدیرعمومی بورد تصحیح کتب درسی و آثار علمی

دیزاین صمد صبا و سیدکاظم کاظمی

سال چاپ: ۱۳۹۹ هجری شمسی

تیراژ: ۱۰۰۰

چاپ: اول

ویب سایت: www.tveta.gov.af

ایمیل: info@tveta.gov.af

حق چاپ برای اداره تعلیمات تخنیکی و مسلکی محفوظ است.

سرود ملی

دا وطـــن افغانســـتان دی — دا عـــزت د هـــر افغـــان دی

کـور د سـولې کـور د تـورې — هـر بچـی یـې قهرمـان دی

دا وطـــن د ټولـــو کـــور دی — د بلوڅـــــو، د ازبکـــــو

د پښـــتون او هـــزاره وو — د ترکمنـــــو، د تاجکـــــو

ورسـره عـــرب، ګوجـــر دي — پامیریـــان، نورســـتانیان

براهـــوي دي، قزلبـــاش دي — هـــم ایمـــاق، هـــم پشـــه‌یان

دا هیـــواد بـــه تـــل ځلیـــږي — لکـــه لمـــر پـــر شـــنه آســـمان

پـــه ســـینه کـــې د آســـیا بـــه — لکـــــه زړه وی جاویـــــدان

نـــوم د حـــق مـــو دی رهبـــر — وایـــو الله‌اکبر وایـــو الله‌اکبر

پیام ادارهٔ تعلیمات تخنیکی و مسلکی

استادان نهایت گرامی و محصلان ارجمند!

تربیت نیروی بشری ماهر، متخصص و کارآمد از عوامل کلیدی و انکارناپذیر در توسعهٔ اقتصادی و اجتماعی هر کشور محسوب می‌گردد و هر نوع سرمایه‌گذاری بزرگ در بخش‌های مختلف اقتصادی نیازمند به پلان‌گذاری و سرمایه‌گذاری در بخش نیروی بشری و توسعهٔ منابع این نیرو می‌باشد. بر مبنای این اصل و بر اساس فرمان شماره ۱۱ مقام عالی ریاست جمهوری اسلامی افغانستان به تاریخ ۱۳۹۷/۲/۱ ادارهٔ تعلیمات تخنیکی و مسلکی از بدنهٔ وزارت معارف مجزا و فصل جدیدی در بخش عرضه خدمات آموزشی در کشور گشوده شد.

اداره تعلیمات تخنیکی و مسلکی به‌عنوان متولی و مجری آموزش‌های تخنیکی و مسلکی در کشور محسوب می‌شود که در چارچوب استراتژی ۵ ساله خویش دارای چهار اولویت مهم که عبارت‌اند از افزایش دسترسی عادلانه و مساویانه فراگیران آموزش‌های تخنیکی و مسلکی در سطح کشور، بهبود کیفیت در ارائه خدمات آموزشی، یادگیری مادام‌العمر و پیوسته و ارائه آموزش نظری و عملی مهارت‌ها به‌طور شفاف، کم‌هزینه و مؤثر که بتواند نیاز بازار کار و محصلان را در سطح محلی، ملی و بین‌المللی برآورده کند، می‌باشد.

این اداره که فراگیرترین نظام تعلیمی کشور در بخش تعلیمات تخنیکی و مسلکی است، تلاش می‌کند تا در حیطهٔ وظایف و صلاحیت خود زمینهٔ دستیابی به هدف‌های تعیین‌شده را ممکن سازد و جهت رفع نیاز بازار کار، فعالیت‌های خویش را توسعه دهد.

نظام اجتماعی و طرز زندگی در افغانستان مطابق به احکام دین مقدس اسلام و رعایت تمامی قوانین مشروع و معقول انسانی عیار است. ادارهٔ تعلیمات تخنیکی و مسلکی جمهوری اسلامی افغانستان نیز با ایجاد زمینه‌های لازم برای تعلیم و تربیت جوانان و نوجوانان مستعد و علاقه‌مند به حرفه‌آموزی، ارتقای مهارت‌های شغلی در سطوح مختلف مهارتی، تربیت کادرهای مسلکی و حرفوی و ظرفیت‌سازی تخصصی از طریق انکشاف و ایجاد مکاتب و انستیتوت‌های تخنیکی و مسلکی در سطح کشور با رویکرد ارزش‌های اسلامی و اخلاقی فعالیت می‌نماید.

فلهذا جهت نیل به اهداف عالی این اداره که همانا تربیهٔ افراد ماهر و توسعهٔ نیروی بشری در کشور می‌باشد؛ داشتن نصاب تعلیمی بر وفق نیاز بازار کار امر حتمی و ضروری بوده و کتاب درسی یکی از ارکان مهم فرایند آموزش‌های تخنیکی و مسلکی محسوب می‌شود، پس باید همگام با تحولات و پیشرفت‌های علمی نوین و مطابق نیازمندی‌های جامعه و بازار کار تألیف و تدوین گردد و دارای چنان ظرافتی باشد که بتواند آموزه‌های دینی و اخلاقی را توأم با دست‌آوردهای علوم جدید با روش‌های نوین به محصلان انتقال دهد. کتابی را که اکنون در اختیاردارید، بر اساس همین ویژگی‌ها تهیه و تدوین گردیده است.

بدین‌وسیله، صمیمانه آرزومندیم که آموزگاران خوب، متعهد و دلسوز کشور با خلوص نیت، رسالت اسلامی و ملی خویش را ادا نموده و نوجوانان و جوانان کشور را به‌سوی قله‌های رفیع دانش و مهارت‌های مسلکی رهنمایی نمایند و از محصلان گرامی نیز می‌خواهیم که از این کتاب به‌درستی استفاده نموده، در حفظ و نگهداشت آن سعی بلیغ به خرج دهند. همچنان از مؤلفان، استادان، محصلان و اولیای محترم محصلان تقاضا می‌شود نظریات و پیشنهادات خود را در مورد این کتاب از نظر محتوا، ویرایش، چاپ، اشتباهات املایی، انشایی و تایپی عنوانی ادارهٔ تعلیمات تخنیکی و مسلکی کتباً ارسال نموده، امتنان بخشند.

در پایان لازم می‌دانیم در جنب امتنان از مؤلفان، تدوین‌کنندگان، مترجمان، مصححان و تدقیق کنندگان نصاب تعلیمات تخنیکی و مسلکی از تمامی نهادهای ملی و بین‌المللی که در تهیه، تدوین، طبع و توزیع کتب درسی زحمت‌کشیده و همکاری نموده‌اند، قدردانی و تشکر نمایم.

ندیمه سحر

رئیس ادارهٔ تعلیمات تخنیکی و مسلکی جمهوری اسلامی افغانستان

فهرست

عنوان | صفحه

مقدمه

مضمون مدیریت سیستم با هدف تربیت نیروهای مسلکی با قابلیت پلان‌ریزی، دیزاین و تطبیق راه حل‌های مناسب برای نیازمندی‌های تکنالوژی معلوماتی در جامعه تنظیم شده است. موضوعات بر اساس محیط سرورهای لینوکس ارائه شده است. مطالب عمدۀ این کتاب در مورد عیارسازی سرویس‌های اساسی شبکه و تطبیق آن در محیط لینوکس می‌باشد. در شروع هر فصل، اهدافی که محصل بعد از خاتمۀ فصل باید قادر به فهم و انجام آن باشد، بیان شده است. ابتدا مفاهیم اولیه و نظری موضوع هر فصل تشریح شده است تا محصل فهم کلی در مورد موضوع پیدا کند و در ادامه مباحث تخنیکی و عملی آورده شده است. در مواردی که مطالب عمده عملی می‌باشد، تلاش شده تا یک سناریوی واقعی شبکه مطرح شود و مرحله به مرحله توضیح داده شده تا محصل به آسانی بتواند پروسۀ تطبیق آن را به خاطر بسپارد. در اخیر فصل نیز علاوه بر خلاصۀ همان فصل، سوالات و فعالیت‌های عملی آورده شده تا محصل بتواند خود را ارزیابی کند. این سوالات و فعالیت‌ها با اهداف اولیۀ فصل در مطابقت است. بعضی از سوالات و فعالیت‌ها انگیزشی است و برای تشویق محصل به تحقیق، آورده شده تا روحیۀ تحقیق در آنها تقویت شود.

اینجانب بسیار خرسندم که فرصت یافتم تا بخشی از دانش خود را در اختیار محصلان نسل جوان این وطن قرار دهم و از خداوند متعال سپاسگزارم که توفیق نشر علم را عطا نمود. در تألیف این کتاب علی‌رغم زمان کم برای آماده‌سازی آن، حد اکثر تلاش خویش را نموده‌ام تا کیفیت مطالب ارائه‌شده مناسب حال و سویۀ محصلان بوده باشد. از آنجایی که این کتاب ویرایش اول است، ممکن است خالی از اشتباه نباشد، لذا از نظریات نیک شما خوانندگان گرامی استقبال می‌کنم تا در هرچه بهترشدن این کتاب ما را یاری رسانید.

در فرجام نیز از تمام کسانی که زمینۀ تألیف کتاب‌های مسلکی برای محصلان عزیز کشورمان را فراهم نمودند، استادان و همکاران گرامی در پوهنتون‌های کابل، مرکز تعلیمات مسلکی و دوستانی که با مشوره و نظریات نیک خویش مرا در ترتیب و تنظیم این اثر یاری نمودند، اظهار سپاس و امتنان می‌نمایم.

هدف کلی کتاب

آشنایی با اسکریپت نویسی، مدیریت حساب کاربر، نصب و اعیارسازی سرور های چون (DNS, File Server, DHCP, FTP, Web Server).

فصل اول

توزیع‌های لینوکس

هدف کلی: محصلان با سیستم‌عامل‌های مخصوص سرور آشنایی حاصل کنند.

اهداف آموزشی: در پایان این فصل محصلان قادر خواهند شد تا:

۱. سیستم‌عامل لینوکس سرور را به‌صورت جامع تشریح نمایند.
۲. منابع مورد نیاز جهت نصب سیستم‌عامل لینوکس را بیان دارند.
۳. روش‌های نصب سیستم‌عامل لینوکس سرور را تشریح نمایند.

به مجموعه‌یی از برنامه‌ها، بسته‌ها و امکانات مدیریتی و ویژگی‌هایی که در یک کرنل لینوکس قرار گرفته است، یک توزیع لینوکس گفته می‌شود. کرنل تنها چیزی است که بین تمام توزیع‌های لینوکس مشترک است و همهٔ آنها از این نظر لینوکس هستند. توزیع‌های مختلفی از لینوکس وجود دارند. این توزیع‌ها از جهت‌های مختلف متفاوت هستند که سه جهت از مهمترین آنها عبارتند از:

۱. هدف
۲. تنظیمات[۱] و بسته‌های نرم‌افزاری
۳. مودل پشتیبانی

از نظر هدف، بعضی از توزیع‌های لینوکس مخصوص سرورها معرفی شده‌اند و بعضی دیگر برای دسکتاپ‌ها ساخته شده‌اند و بعضی نیز برای مقاصد خاص مثل کلاینت‌ها در نظر گرفته شده‌اند. اما بیشتر از توزیع‌های لینوکس مخصوص سرور استفاده می‌شود و از لینوکس برای دسکتاپ کمتر استفاده می‌شود.

تفاوت عمدهٔ دیگر در تنظیمات می‌باشد، بعضی از توزیع‌ها فایل‌های تنظیمات را در آدرس‌های متفاوتی در سیستم ذخیره می‌کنند و برخی دیگر در یک مسیر آنها را ذخیره می‌کنند، همچنان تفاوت عمده از نظر بسته (package) وجود دارد که منظور از بسته، فایل‌های باینری فایل‌های نصب برنامه‌ها می‌باشد. و ابزارهای مدیریتی مختلفی برای مدیریت بسته‌ها در توزیع‌های مختلف وجود دارد.

تفاوت سوم از نظر پشتیبانی می‌باشد، درحالی‌که بعضی از توزیع‌ها مثل Debian, CentOS و Fedora توسط برنامه‌نویسان و توسعه‌دهندگان داوطلب پیش برده می‌شوند، توزیع‌های دیگری مثل Red Hat Enterprise Linux و Ubuntu توسط ارئه‌دهندگان تجاری پشتیبانی می‌شوند. به‌طور کلی دو گروپ از توزیع‌ها وجود دارند (Soyinka,۲۰۱۶).

۱.۱ توزیع‌های Red Hat

Red Hat یکی از رایج‌ترین توزیع‌های با پشتیبانی تجاری لینوکس است که نسخه‌های مختلفی دارد. دو نسخهٔ آن از همه معروف‌تر است، نسخهٔ RHEL و نسخهٔ RHELAP؛ تفاوت آنها در تعداد CPUهایی است که پشتیبانی می‌کنند که نسخهٔ اول حد اکثر تا دو CPU را پشتیبانی می‌کند و نسخهٔ دوم به تعداد نامحدودی CPU را پشتیبانی می‌کند. بسته‌های نرم‌افزاری این نوع توزیع‌ها RPM می‌باشد. توزیع‌های دیگری مثل CentOS, Fedora از این نوع هستند. این توزیع‌ها تغییریافتهٔ RHEL هستند با این تفاوت که به‌صورت رایگان و بدون پشتیبانی ارائه شده‌اند. Fedora بیشتر برای تست ویژگی‌های جدید RHEL معرفی شد و از آن در سرورها استفاده می‌شود.

[۱] - configuration

۱.۲ توزیع‌های Debian

این توزیع یک توزیع رایگان است که توسط تعدادی زیادی از توسعه‌دهندگان و کاربران فعال حمایت می‌شوند. این توزیع در سال ۱۹۹۳ شروع به کار کرد. بسته‌های نرم‌افزاری مخصوص این توزیع‌ها به نام dpkg شناخته می‌شوند و نرم‌افزارها و ابزارهای زیادی برای این توزیع ساخته شده است. Ubuntu نیز از این توزیع گرفته شده است (Negus,۲۰۱۰).

۱.۲.۱ انتخاب توزیع مناسب

انتخاب توزیع مناسب برای یک سازمان به عوامل مختلفی بستگی دارد (بودجه، مهارت‌های کارمندان و نیازمندی‌های سازمان) به‌جز Red Hat که برای پشتیبانی نیازمند پرداخت هزینه می‌باشد، سایر توزیع‌های معرفی‌شده در فوق پشتیبانی رایگان دارند و شما می‌توانید آنها را به‌صورت رایگان بدون پرداخت هزینه برای لایسنس تهیه نمایید. اگر سازمان شما افراد مسلکی به اندازهٔ کافی ماهر یا قوی در زمینهٔ IT ندارد می‌توانید از توزیع‌هایی استفاده کنید که پشتیبانی فنی خوبی در ازای دریافت هزینهٔ اندک دارند؛ مثل: Red Hat اما در اکثر مواقع نیاز به آن نیست زیرا پشتیبانی‌های رایگان خوبی نیز برای سایر توزیع‌ها در انترنت موجود است و کاربران و توسعه‌دهندگان مختلفی همزمان برای ازبین‌بردن مشکلات هر یک از توزیع‌ها تلاش می‌کنند. در نهایت می‌توانید برای انتخاب توزیع مناسب با سلیقهٔ شخصی خود آنها را امتحان کنید.

دستورات و تشریحات این کتاب بیشتر بر اساس توزیع‌های شبه Red Hat یعنی Red Hat, Fedora, CentOS است. زیرا این سیستم‌عامل‌ها بیشتر برای سرورها مورد توجه می‌باشند. یعنی مطالبی را که در این کتاب بیان می‌شود، شما باید بتوانید در توزیع‌های مختلف از این دست به کار ببرید، فرقی نمی‌کند که کدام نسخه و کدام توزیع را شما ممکن است استفاده کنید. مثال‌های عملی کتاب بر اساس CentOS7 است (Soyinka,۲۰۱۶).

۱.۲.۲ انتخاب سخت‌افزار مناسب

انتخاب سخت‌افزار مناسب برای هر توزیع از چوکات این کتاب خارج است و ما به‌طور عموم توصیه می‌کنیم که یک سیستم با مشخصات حد اقلی را برای مصارف لابراتواری تهیه نمایید، یک کمپیوتر با حد اقل GB۲ حافظهٔ RAM و GB۲۰ فضای ذخیره‌سازی و یک cpu با قدرت محاسباتی حد اقل ۱.۵ GHz برای راه‌اندازی یک سیستم‌عامل توزیع لینوکس مخصوص سرور کافی است.

یک نکتهٔ دیگر در مورد سخت‌افزار وجود دارد که ممکن است مشکل‌ساز شود، درحالی‌که لینوکس از بسیاری از سخت‌افزارها پشتیبانی می‌کند، بعضی از سخت‌افزارها ممکن است درایوری برای نصب آنها در لینوکس وجود نداشته باشد (مثل بعضی کارت‌های توسعه یا کارت شبکهٔ بی‌سیم و...)، قبل از انتخاب توزیع لینوکس باید از این نکته اطمینان حاصل کنید. برای این کار لستی به نام HCL وجود دارد و وب‌سایت‌هایی در انترنت موجود است که شامل این لست‌ها می‌باشند. این لست شامل سخت‌افزارهایی است که توسط توزیع لینکوس پشتیبانی می‌شود. در جدول زیر لستی از این وب‌سایت‌ها آمده است (Soyinka,۲۰۱۶).

جدول ۱-۱ لست وب‌سایت‌هایی که معلومات HCL برای توزیع‌های مختلف لینوکس را دارند

سیستم‌عامل	آدرس سایت
Red Hat, CentOS, and Fedora	https://access.redhat.com/ecosystem
Ubuntu	http://www.ubuntu.com/certification
Debian, but also relevant for Ubuntu	http://kmuto.jp/debian/hcl/wiki/
All distribution	www.linuxquestions.org/hcl/index.php

۱.۳ منابع مورد نیاز جهت نصب سیستم‌عامل سرور

از آنجایی که Fedora و CentOS بر اساس RHEL بنا نهاده شده‌اند، لذا منابع سخت‌افزاری مورد نیاز آنها نیز یکسان است و این برای سیستم‌عامل دسکتاپ و سرور فرقی نمی‌کند. سخت‌افزار مورد نیاز برای یک تجربهٔ خوب با این توزیع‌ها به‌صورت گرافیکی شامل موارد زیر است هرچند با سخت‌افزار کمتر از آن هم می‌توان یک سیستم‌عامل لینوکس را اجرا کرد.

پردازنده (CPU): یک CPU ۴۰۰ MHz Pentium حد اقل چیزی است که برای نصب گرافیکی لازم است. برای بیشتر کاربرها (x۸۶) CPU ۳۲-bit کفایت می‌کند اما اگر می‌خواهید که از سیستم‌عامل خود برای مقاصد مجازی‌سازی نیز استفاده کنید بهتر است از (x۸۶-۶۴) CPU ۶۴ استفاده کنید.

۱. **حافظهٔ RAM**: حد اقل ۱ GB ولی برای تجربهٔ بهتر گرافیکی ۳-۲ GB توصیه می‌شود.
۲. **CD/DVD یا USB**: برای نصب سیستم‌عامل ضرورت به یک حافظهٔ جانبی مثل DVD یا فلش مموری می‌باشید که از قبل bootable شده باشند. فضای لازم برای آنها ۷ GB توصیه می‌شود.
۳. **فضای دسک**: حد اقل ۱۰ GB فضای خالی هارددسک برای نصب لازم است؛ البته فضای واقعی مورد نیاز بستگی به سناریوی نصب شما دارد؛ مثلاً: اگر سرور را به‌صورت حد اقل بدون GUI نصب کنید فقط ۶۰۰ MB فضا لازم است و اگر به‌طور کامل نصب کنید ۷ GB فضا را خواهد گرفت و مقداری هم فضا برای ذخیره‌کردن فایل‌های مربوط به کاربران و سایر فعالیت‌های نورمال سیستم‌عامل مورد نیاز است.
۴. **سخت‌افزارهای خاص**: بعضی از توزیع‌های لینوکس نیازمند ویژگی‌های خاص سخت‌افزاری می‌باشند؛ مثلاً برای نصب لینوکس با قابلیت مجازی‌سازی KVM، CPU شما باید قابلیت مجازی‌سازی AMD-V یا Intel-VT را داشته باشد.

۱.۴ روش‌های نصب سیستم‌عامل سرور

امروزه پروسهٔ نصب سرور لینوکس آسان شده است.

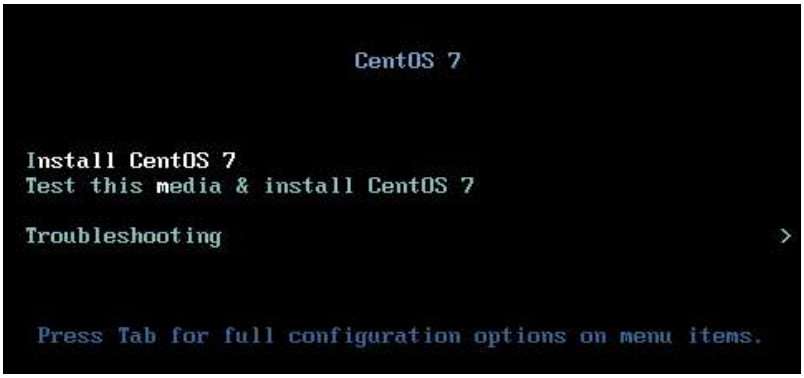
- **نصب از طریق Live CD**: منظور از Live CD در اصل CD، DVD ویا فایل ISOیی است که ممکن است بر روی یک فلش مموری یا CD/DVD نوشته شده باشد و خاصیت bootable را داشته باشد. یعنی بدون سیستم‌عامل اجرا شود. در این روش نصب ضرورتی به هارددسک برای اجرای سیستم‌عامل لینوکس نخواهد بود و فایل‌های سیستم‌عامل از طریق boot وارد حافظهٔ RAM شده و مستقیماً از روی RAM اجرا می‌شوند بدون نیاز به هارددسک. در این روش هر بار برای اجرای سیستم‌عامل باید CD/DVD را در کمپیوتر قرار دهید.
- **نصب از طریق یک مدیا:** منظور از مدیا حافظهٔ خارجی است که شامل فایل‌های اولیهٔ سیستم‌عامل لینوکس برای نصب باشد؛ مثل یک DVD یا USB درایو. بیشتر توزیع‌های لینوکس را می‌توانید از طریق DVD نصب کنید، معمولاً علاوه بر سیستم‌عامل شامل بسته‌های نرم‌افزاری نیز می‌باشند که می‌توانید در هنگام نصب آنها را نیز انتخاب و نصب نمایید. در این روش ابتدا سیستم‌عامل از روی DVD در هارددسک کاپی شده و سپس نصب می‌شود.
- **نصب در ابعاد وسیع:** اگر بخواهید که سیستم‌عامل را همزمان در ده‌ها یا صدها سیستم کمپیوتر نصب کنید، دیگر روش‌های قبلی، مناسب نیست و باید از روش نصب از طریق شبکه استفاده کرد.
- **نصب در محیط مجازی:** روش نصب سیستم‌عامل در داخل یک سیستم‌عامل دیگر به کمک یک نرم‌افزار مدیریت محیط مجازی (VMM[1]) یکی از روش‌های دیگر نصب به شمار می‌رود.

در این کتاب، روش دوم نصب، یعنی نصب از طریق یک DVD یا USB درایو را بیان می‌کنیم زیرا یکی از روش‌های رایج نصب سیستم‌عامل سرور به شمار می‌رود. در ادامه مراحل نصب سیستم‌عامل CentOS ۷ را آموزش می‌دهیم.

۱. بعد از دانلود آخرین نسخهٔ CentOS با استفاده از لینک‌های بالا ویا با استفاده از صفحهٔ رسمی CentOS download، آن را روی یک DVD رایت کنید ویا با استفاده از نرم‌افزار LiveUSB Creator به نام Unetbootin، یک USB stick با قابلیت بوت بسازید.
۲. بعد از این‌که میدیای bootable نصب را، ساختید، DVD یا USB را داخل درایو مناسب در سیستم خود قرار دهید؛ کمپیوتر را روشن کنید؛ بخش bootable خود را انتخاب کنید و اولین خط فرمان CentOS7 ظاهر خواهد شد. در خط فرمان، Install CentOS7 را انتخاب کنید و کلید [Enter] را بفشارید.

---

[1]- Virtual Machine Manager

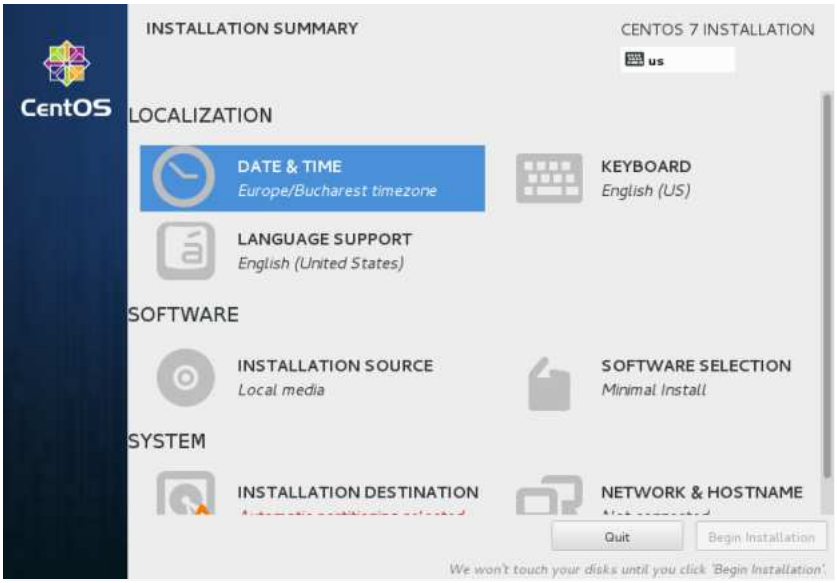


شکل(۱-۱) مرحلهٔ اولیهٔ نصب

۳. سیستم شروع به لودکردن میدیای نصب، می‌کند و یک صفحهٔ خوش‌آمدگویی ظاهر می‌شود. در این مرحله باید زبان نصب سیستم‌عامل را انتخاب کنید.

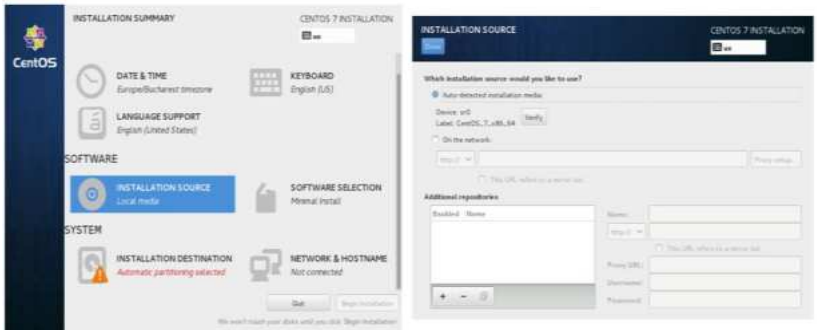
شکل(۱-۲) انتخاب زبان نصب

۴. مرحلهٔ بعد، باید تنظیمات بیشتری را انجام دهید، تنظیمات به سه کتگوری تقسیم شده است، تنظیمات محلی سیستم (شامل تنظیم تاریخ سیستم، زبان‌های سیستم‌عامل، انتخاب نوع صفحهٔ کلید)، تنظیمات نرم‌افزارها (نصب نرم‌افزارهای مورد نیاز) و تنظیمات مربوط به سیستم‌عامل (انتخاب محل نصب سیستم‌عامل و پارتیشن‌بندی فایل‌سیستم و نهایتاً تنظیمات شبکه). که شما از تنظیم تاریخ سیستم شروع کنید و مطابق آنچه که در شکل زیر نمایش داده شده است در Date&Time کلیک کنید، سپس زبان و صفحهٔ کلید را نیز انتخاب کنید.



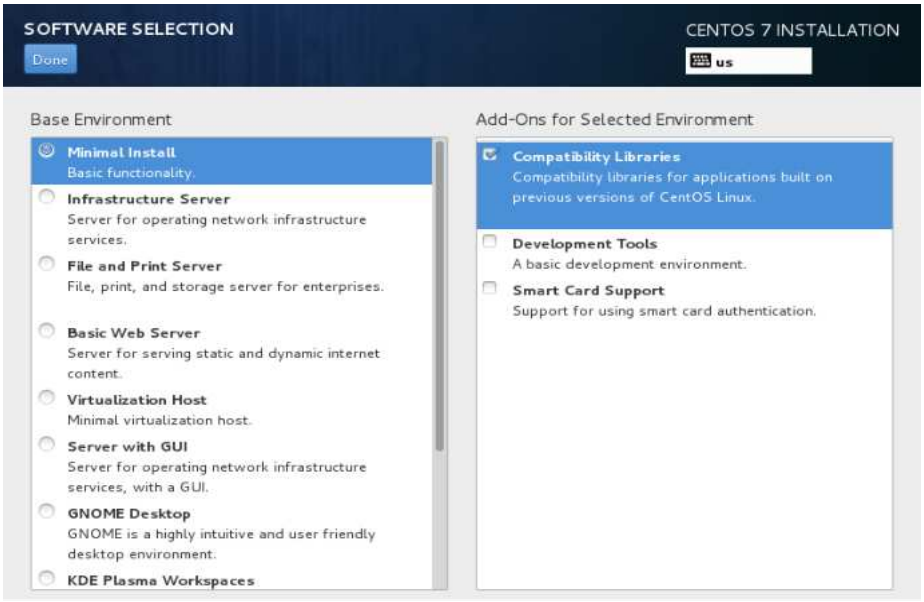
شکل(۱-۳) تنظیمات قبل از نصب

۵. در مرحلهٔ بعد شما می‌توانید با استفاده از دیگر منابع نصب از میدیای DVD یا USB درایو و حتی از طریق شبکه با استفاده از پروتوکول‌های HTTP، HTTPS، FTP نرم‌افزارهای مورد نیاز خود را نصب کنید. در این بخش شما گزینهٔ Auto-detected installation media را انتخاب کرده و ادامه دهید.



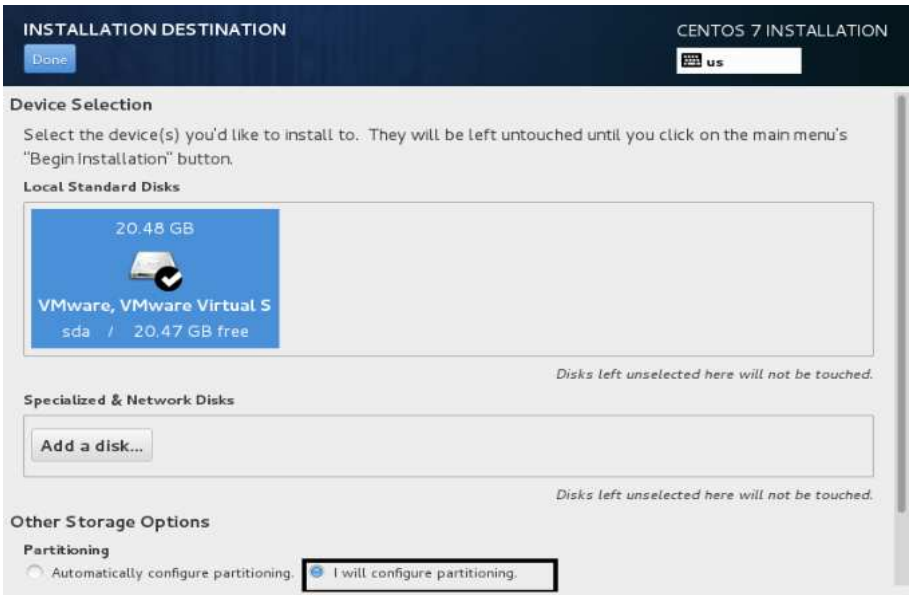
شکل(۱-۴) انتخاب منبع نصب نرم‌افزارها

۶. در مرحلهٔ بعد شما می‌توانید نرم‌افزارهای مورد نیاز سرور خود را با کلیک در گزینهٔ software selection انتخاب کنید تا همراه سیستم‌عامل نصب شوند. در این بخش نظر به نوع وظیفه‌یی که این سرور می‌تواند داشته باشد، محیط‌هایی از قبل در نظر گرفته شده است که هر محیط شامل مجموعهٔ نرم‌افزارهای مرتبط با نوع سرور می‌باشد. اگر هنوز مطمئن نیستید که از سیستم‌عامل به‌عنوان چه نوع سروری استفاده می‌کنید، می‌توانید گزینهٔ Minimal Install را به همراه Compatibility Libraries به‌عنوان Add-ons انتخاب کنید، که در این دسکتاپ گرافیکی دیگر نصب نمی‌شود و بعد از آن می‌توانید بسته‌های دیگری را که مورد نیازتان است با استفاده از دستور yum groupinstall اضافه کنید.



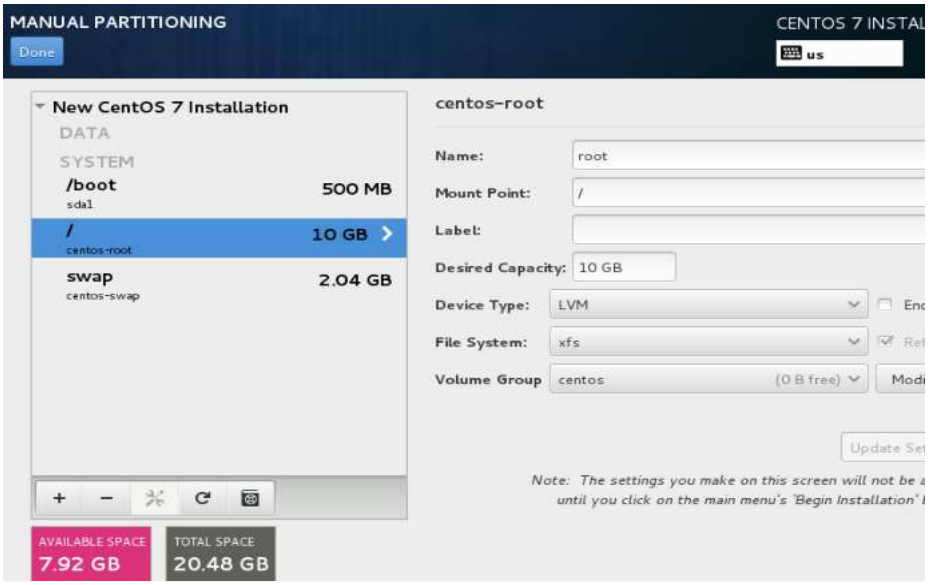
شکل(۱-۲) انتخاب نرم‌افزارها

۷. اکنون زمان پارتیشن‌بندی هارددرایو شما می‌باشد. در مینوی Installation Destination کلیک کنید، دسک مورد نظرتان را انتخاب کنید و سپس I will configure partitioning را انتخاب نمایید.



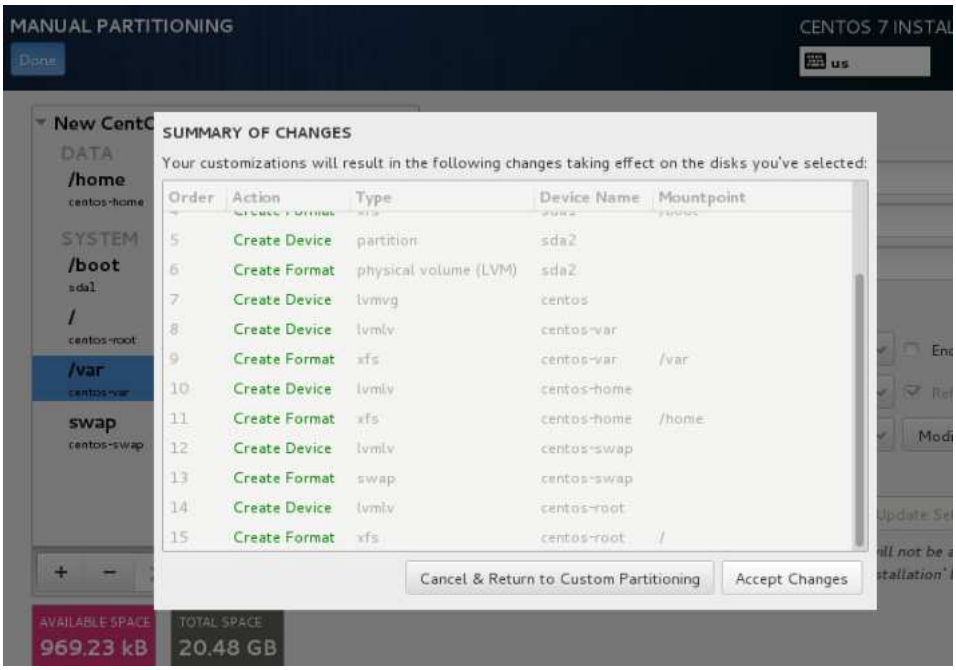
شکل(۱-۶) انتخاب محل نصب سیستم‌عامل

۸. در صفحهٔ بعد، LVM یا Logical Volume Manager را به‌عنوان Layout پارتیشن انتخاب کنید و سپس روی Click here to create them automatically کلیک کنید، گزینه‌یی که سه پارتیشن سیستمی با فورمت XFS خواهد ساخت، فضای هارددسک شما را به‌صورت خودکار تقسیمات می‌کند و تمامی LVS در یک Volume Group بزرگ به نام CentOS، ایجاد می‌شوند.



شکل (۱-۷) پارتیشن‌بندی پیش‌فرض

۹. اگر با انجام Layout پارتیشن پیش‌فرض به‌صورت خودکار، موافق نیستید، می‌توانید ترکیب پارتیشن‌بندی خود را اضافه، اصلاح یا تغییر سایز دهید و وقتی کارتان تمام شد، دکمهٔ Done و Accept Changes در خط فرمان Summary of Changes را بفشارید.



شکل (۱-۳) تأیید پارتیشن‌بندی هارددسک

توجه: برای کاربرانی که هارددسک‌هایی با اندازهٔ بیش از ۲ ترابایت دارند، نرم‌افزار نصب به‌صورت خودکار جدول پارتیشن بندی را به GPT تبدیل خواهد کرد.

۱۰.مرحلهٔ بعد، تنظیم‌کردن hostname سیستم و فعال‌کردن شبکهٔ آن است. در Network & Hostname کلیک کنید و FQDN یا Fully Qualified Domain Name سیستم خود را در فیلد Hostname تایپ کنید سپس رابط شبکهٔ خود را فعال کنید، دکمهٔ Ethernet را که در بالا قرار دارد، روی ON قرار دهید. اگر یک DHCP server در شبکهٔ خود دارید، به‌صورت خودکار تنظیمات شناسایی خواهد شد.

شکل (۱-۴) تنظیمات شبکهٔ سیستم‌عامل

۱۱.اگر سیستم شما به‌عنوان یک سرور است، بهتر است تنظیمات شبکهٔ استاتیک در NIC انترنت را با کلیک روی دکمهٔ Configure انجام دهید و روی دکمهٔ Save کلیک کنید. کارت Ethernet را با سویچ‌کردن دکمه به OFF و ON غیر فعال و فعال کنید و سپس Done را بفشارید تا تنظیمات اعمال شوند و به مینوی اصلی بازگردید.

۱۲.اکنون زمان شروع پروسهٔ نصب با فشردن دکمهٔ Begin Installation و انتخاب یک پس‌ورد مغلق برای حساب کاربر root است.

۱۳.بعد از انتخاب پس‌ورد root نوبت به ایجاد یک حساب کاربر غیر از root می‌رسد، برای این کار در User Creation کلیک کنید. با انتخاب گزینهٔ Make this user administrator این کاربر تمام صلاحیت‌های root را از طریق دستور sudo خواهد داشت. پس از آن روی Done کلیک کنید و به مینوی اصلی برگردید و تا اتمام پروسهٔ نصب منتظر بمانید.

۱۴.بعد از این‌که پروسهٔ نصب تمام شد، نرم‌افزار نصب یک پیام موفقیت‌آمیز روی صفحه نشان خواهد داد و از شما می‌خواهد تا برای استفاده از سیستم خود، آن را راه‌اندازی کنید.

اکنون شما آخرین نسخهٔ CentOS را روی دستگاه جدید خود، نصب کرده‌اید. میدیای نصب را خارج کنید و کمپیوترتان را راه‌اندازی کنید بنابراین می‌توانید به محیط کاری و جدید CentOS ۷ وارد شوید و کارهای دیگر سیستم از قبیل به‌روزرسانی سیستم‌تان و نصب دیگر نرم‌افزارهای مورد نظرتان را انجام دهید.

خلاصهٔ فصل اول

- به مجموعه‌یی از برنامه‌ها، بسته‌ها و امکانات مدیریتی و ویژگی‌هایی که در یک کرنل لینوکس قرار گرفته است، یک توزیع لینوکس گفته می‌شود.
- توزیع‌ها از جهت هدف، تنظیمات و بسته‌های نرم‌افزاری و نوع پشتیبانی متفاوت هستند. دو توزیع عمده برای لینوکس وجود دارند (توزیع‌های Red Hat و توزیع‌های Debian)
- ایدهٔ اصلی اجتماع نرم‌افزار آزاد بر آزاد بودن بود، آزادی نرم‌افزار به معنای رایگان بودن آن نیست، بلکه بدین معناست که هر شخص آزاد است تا نرم‌افزار متن باز را پس از تغییرات مجدداً به همراه سورس (Code) جدید در اختیار دیگران قرار دهد و آن را بسته نکند، بستن نرم‌افزار به معنای آن است که سورس (Code) آن را در اختیار دیگران قرار ندهید و فقط فایل نصب یا فایل باینری آن را در اختیار دیگران قرار دهید.
- سیستم‌عامل سرور به چهار روش قابل نصب است، از طریق Live CD/DVD، Media، نصب در محیط مجازی، و نصب در ابعاد وسیع از طریق شبکه.

سوالات و فعالیت های فصل اول

۱. توزیع لینوکس چیست؟ شباهت‌ها و تفاوت‌های توزیع‌های مختلف را بیان کنید.

۲. چطور می‌توان یک توزیع مناسب برای سرور لینوکس خود انتخاب کرد؟

۳. روش‌های نصب سیستم‌عامل لینوکس را نام برده و تشریح کنید.

۴. فرق یک سیستم‌عامل سرور با یک سیستم‌عامل معمولی در چیست؟

فعالیت ها

۱. به جدول ۱ این فصل مراجعه کنید و معلومات HCL را در مورد توزیع‌های Red Hat , Centos مشاهده کنید.

۲. با مراجعه به منابع مختلف از جمله انترنت چهار نمونه از تفاوت‌های عمده بین دو توزیع Red Hat و Debian را پیدا کنید.

۳. فایل نصب سیستم‌عامل Fedora را از سایت رسمی آن دانلود کرده و مراحل نصب آن را بر روی یک ماشین مجازی (virtual machine) مثل VMware ویا Virtual Box پیش ببرید. در موقع نصب سرویس‌های مخصوص سرور مثل DNS، DHCP و... را نیز انتخاب کنید تا نصب شود.

۴. در مورد سایر توزیع‌های جدیدی که مخصوص سرور می‌باشند معلومات کسب کنید.

فصل دوم

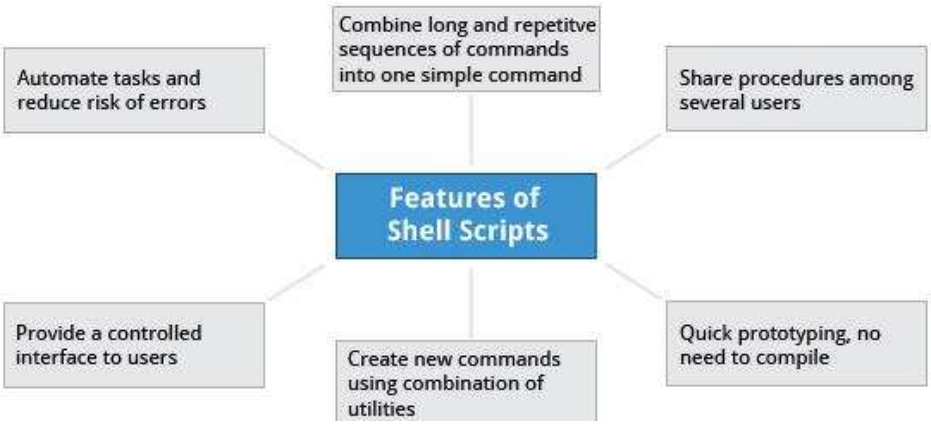
Shell Script در لینوکس

هدف کلی: آشنایی با اسکرِپت‌نویسی و استفاده از آن برای مدیریت سرور.

اهداف آموزشی: در پایان این فصل محصلان قادر خواهند شد تا:

۱. قابلیت‌های Bash Shell Scripting را معرفی کنند.
۲. ساختار دستورات اسکرِپتی را تشریح کنند.
۳. میتودها و دستورات پرکاربرد را توضیح دهند.
۴. فکشن (Function) را تشریح کنند.
۵. دستورات شرطی (If Statements) را به کار برده بتوانند.
۶. محاسبات ریاضیاتی را انجام داده بتوانند.
۷. دستورات تکرار و حلقه‌ها (loops) به کار ببرند.

فرض کنید که شما یک مدیر شبکه هستید و از شما به‌طور عاجل خواسته شده است تا برای ۱۰۰۰ نفر از کارمندان حساب کاربر برای دسترسی به سرور ایجاد کنید، برای این کار به شما فقط یک فایل شامل مشخصات کارمندان داده شده است. یک راه حل این است که از روی لست برای تک تک نام‌هایی که داده شده، دستور مربوط به ایجاد یک حساب کاربر جدید را اجرا کنید و این کار را تا آخرین نفر تکرار کنید، واضح است که چنین کاری درست است که عملی به نظر می‌رسد ولی منطقی نیست، زیرا تکرار یک دستور آن هم ۱۰۰۰ بار منطقی نیست و از طرف دیگر وقت‌گیر است. باید راه حل آسان‌تری وجود داشته باشد. این راه حل آسان استفاده از shell script است، منظور از shell script در اصل مجموعه‌یی از دستورات shell لینوکس است که با یک هدف خاص در یک فایل اجرایی ذخیره شده است و علاوه بر آن یک ویژگی مهم shell script استفاده از قابلیت برنامه‌نویسی در آن است که آن را بسیار قدرتمند ساخته است، به‌طور خلاصه می‌توان گفت shell script جمع دو قابلیت برنامه‌نویسی و دستورات یا دستورهای لینوکس است.



شکل(۲-۱) قابلیت‌های shell script

همان‌طور که در تصویر فوق مشاهده می‌شود، اسکرِپت‌نویسی مزایای بسیاری در محیط لینوکس دارد که برخی از مهم‌ترین آنها عبارتند از:

- ادغام دستورهای طولانی و تکراری؛
- اشتراک‌گذاری اسکرپیت‌ها با سایر کاربران سیستم؛
- پروتوتایپ[1]سازی (ساخت نمونهٔ اولیه) سریع بدون نیاز به کامپایل‌شدن کُد منبع (Source code)؛
- ایجاد دستور جدید با ترکیب‌کردن دستورهای از پیش تعریف‌شده؛
- خودکارسازی وظایف سیستم‌عامل و به حد اقل رساندن خطا.

[1] - prototype

۲.۱ معرفی قابلیت‌های Bash Shell Scripting

۲.۱.۱ تفاوت Shell و Bash

Shell یک نرم‌افزار بسیار سبک است که یک انترفیس (رابط) در اختیار سیستم‌عامل قرار می‌دهد و از آن طریق کاربران سیستم‌عامل می‌توانند دستورات مدنظر خود را به سیستم‌عامل انتقال دهند. شل‌ها یا گرافیکی هستند یا غیر گرافیکی؛ به‌طور مثال: در سیستم‌عامل ویندوز، Shell پیش‌فرض explorer.exe است و در سیستم‌عامل مکینتاش Finder قابلیتی مشابه را در اختیار کاربرانش قرار می‌دهد و در یونیکس/لینوکس هم مثلاً Gnome بخشی از محیط دسکتاپ است که چنین امکانی را فراهم می‌کند.

تمام مثال‌های قبلی جزء شل‌های گرافیکی به حساب می‌آیند که شامل پنجره، مینو، آیکن و ... هستند تا یک GUI (رابط کاربری گرافیکی) در اختیار کاربران عادی قرار دهند تا به‌سادگی بتوانند با سیستم‌عامل ارتباط برقرار سازند؛ اما منظور از Shell در shell script همان رابط غیر گرافیکی یا محیط خط فرمان (CLI) [۱] است، که فقط از طریق تایپ‌کردن دستورات می‌توان با سیستم‌عامل ارتباط برقرار ساخت.

Bash یکی از انواع شل‌های خط فرمان (CLI) است، اما در عین‌حال یکی از محبوب‌ترین شل‌ها است که به‌صورت پیش‌فرض در بسیاری از توزیع‌های سیستم‌عامل گنو/لینوکس گنجانیده شده است که به‌عنوان جایگزینی برای Bourne Shell (یکی از ابتدایی‌ترین انواع شل‌های یونیکس) ابداع شد. (از نمونه‌های شل‌های خط فرمان (CLI) در سیستم‌عامل ویندوز هم می‌توان به cmd.exe یا همان Command Prompt و همچنین PowerShell اشاره کرد).

اسکرِپت در حالت ابتدایی همان دستورات شل لینوکس است که در یک فایل ذخیره شده است. برای درک این سادگی به مثال زیر توجه کنید:

```
find. -name “*.c” -ls
```

در این‌جا هدف این است که نشان دهیم اجرای دستور فوق در خط فرمان (CLI) هیچ تفاوتی در عمل با اجرای فایلی اسکرپتی حاوی دستورات زیر ندارد:

```
#!/bin/bash
find. -name “*.c” -ls
```

در اسکرپیت فوق، خط اول که با علامت !# شروع شده مسیر کامل شل مورد نظر برای اجرای دستورات را مشخص می‌کند.

---

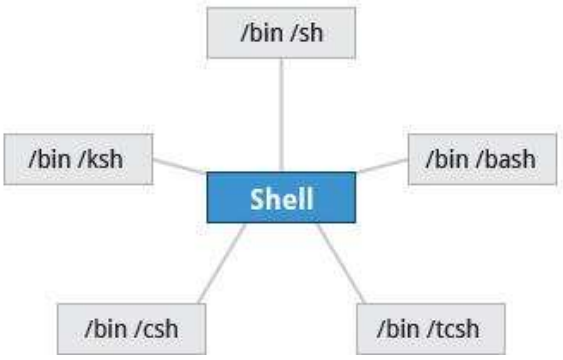
[۱] - command line

۲.۱.۲ انواع shell

در لینوکس تعداد زیادی از Shellها در اختیار کاربران قرار دارد که عبارتند از:

- /bin/sh
- /bin/bash
- /bin/tcsh
- /bin/csh
- /bin/ksh

به خاطر داشته باشید در مسیر /etc/shells می‌توانید به لستی از شل‌های موجود روی سیستم‌عامل دست یافت (Soyinka,۲۰۱۶). بسیاری از کاربران لینوکس از شلی تحت عنوان Bash استفاده می‌کنند اما همان‌طور که در تصویر زیر مشخص است، به غیر از «بش»، گزینه‌های دیگری نیز در اختیار کاربران لینوکس قرار دارد:



شکل(۲-۲) انواع شل‌های لینوکس

اسکرِپت‌های Shell پس از پایان اجرا، Value (قیمت) را اصطلاحاً return (برگشت) می‌کنند؛ هر دستوری که اجرا می‌شود یا موفق است یا ناموفق، اگر موفق باشد قیمت صفر را به سیستم‌عامل بر می‌گرداند و در غیر این صورت قیمت غیر صفر را برگشت می‌دهد. از این قیمت برگشتی می‌توان فهمید که نتیجهٔ یک دستور موفقیت آمیز بوده است یا خیر.

نکته: هر دستور در لینوکس یک نتیجه دارد این نتیجه یا صفر است یا غیر صفر، اگر صفر باشد به معنای اجرای موفق آن است و اگر غیر صفر باشد به معنای اجرای ناموفق آن دستور است. این عدد در یک متغیر محیطی (Environment Variable) تحت عنوان $? ذخیره می‌شود. برای مشاهدهٔ نتیجهٔ یک دستور بعد از اجرای آن می‌توان دستور echo $? را تایپ کرد تا نتیجهٔ آن را نشان دهد.

برای روشن‌تر شدن این مسأله مثالی می‌زنیم؛ دستور ls /etc/passwd را در ترمینال وارد می‌کنیم و در پاسخ /etc/passwd نمایش داده می‌شود چراکه سیستم باموفقیت توانسته این فایل را بیابد و مقدار ۰ را به‌عنوان اجرای موفق نشان می‌دهد. که همیشه در متغیری تحت عنوان $? ذخیره می‌شود و با استفاده از دستور echo می‌توان آن را روی صفحه نمایش داد:

```
$echo $?
```

جهت چک‌کردن این موضوع، دستور زیر را در ترمینال وارد می‌کنیم:

```
$ls /etc/ passwd2
ls: cannot access '/etc/ passwd2 ': No such file or directory
```

درواقع، از آنجا که ما هیچ فایلی تحت عنوان passwd۲ روی سیستم خود نداریم، می‌بینیم که پیامی ظاهر می‌شود که چنین فایلی یافت نشد؛ حال اگر بخواهیم ببینیم که در حال حاضر چه مقداری در متغیر $? ذخیره شده است، مجدداً دستور echo $? را در ترمینال وارد می‌کنیم:

```
$echo $?
2
```

همان‌طورکه پیش از این گفتیم، هر زمانی‌که اجرای دستور باموفقیت همراه نباشد، عددی به غیر از ۰ در نظر گرفته خواهد شد که در اینجا عدد ۲ است.

۲.۲ آشنایی با ساختار اسکرِپت‌نویسی

۲.۲.۱ کرکترهای خاص در اکسرپت‌نویسی

شل لینوکس حساس به کرکترهای خورد و کلان است یعنی کرکترهای خورد و کلان متفاوت هستند و باید به این نکته توجه جدی داشته باشید. همچنان بعضی کرکترها از نظر شل معنای خاصی می‌دهد و در استفاده از آنها باید دقت نمود که به آنها کرکترهای خاص گفته می‌شود. برای این که یک اسکرِپت درست و اصولی باشد، باید اصول، قوانین و ساختار خاصی در نوشتن آن رعایت شود که در جدول زیر برخی از رایج‌ترین کرکترهای خاص در محیط Bash آمده است:

جدول (۲-۱) کرکترهای خاص

کرکتر	کاربرد
#	برای کامنت‌گذاری استفاده می‌شود به غیر از مواقعی که به‌صورت #\ ویا !# در ابتدای یک اسکرِپت استفاده شود.
\	در انتهای یک خط استفاده می‌شود تا ادامهٔ اسکرِپت به خط بعد را اعلام کند.
;	هر دستوری که پس از این علامت قرار گیرد، به‌عنوان یک دستور جدید تلقی خواهد شد.
$	هر چیزی که پس از این علامت قرار گیرد؛ مثلاً($sname) به‌عنوان یک متغیر شناخته می‌شود.

۲.۲.۲ شکستن خطوط طولانی

بعض اوقات اسکرِپت‌نویسان مجبور هستند که زنجیرهایی از دستورها را یکی پس از دیگری اجرا کنند که در چنین حالاتی علامت \ که تحت عنوان Concatenation Operator شناخته می‌شود استفاده شده تا دستورات طولانی را که در چند خط قرار می‌گیرند، به یکدیگر وصل کند.

برای مثال، درصورتی‌که بخواهید فایلی به نام /var/ftp/pub/userdata/custdata/read را از یک سرور فرضی به نام server۱linux.com. به دایرکتری /opt/oradba/master/abc روی یک سرور مثلاً تحت عنوان server۳linux.co.in. کاپی کنید، به‌سادگی و با استفاده از کرکتر \ می‌توان این‌کار را انجام داد:

```
scp abc@server1.linux.com:\
var/ftp/pub/userdata/custdata/read \
abc@server3.linux.co.in:\
/opt/oradba/master/abc/
```

عملگرا (operator) این امکان را برای ما فراهم می‌سازد تا دستورها را در چندین خط مجزا از یکدیگر قرار دهیم که این مسأله باعث خوانایی بیشتر اسکرِپت‌ها می‌شود؛ به‌عبارت دیگر، \ در انتهای هر خط باعث وصل‌شدن خط قبلی با خط بعدی می‌شود و درنهایت می‌توان تمامی آنها را به شکل یک دستور واحد اجرا کرد.

۲.۲.۳ قراردادن چندین دستور در یک خط

گاهی نیاز است تا چندین دستور را در یک خط واحد پیهم بنویسیم. در چنین مواقعی علامت ; برای جدا سازی این دستورها از یکدیگر و اجرای آنها به همان ترتیبی که نوشته شده‌اند، استفاده می‌شود؛ برای مثال، دستور زیر شامل ۳ دستور مجزا از یکدیگر است:

```
$ Make; make install; make clean
```

نکته: حتی اگر اجرای دستور اول با موفقیت همراه نباشد، دستورهای دیگر اجرا خواهند شد. اگر بخواهیم شرطی قرار دهیم که اگر دستور اول -یا دستورهای قبلی- بدون موفقیت اجرا شدند، دستورهای بعدی اجرا نشوند، می‌توان از ساختار زیر استفاده کرد:

```
$make && make install && make clean
```

در دستور فوق، اگر دستور اول اصطلاحاً Fail شود، دستور دوم هرگز اجرا نخواهد شد. در عین‌حال شرایطی هم ممکن پیش آید که ضرورت به این شرط باشد که از بین چند دستور حد اقل یکی از آنها با موفقیت اجرا شوند. در چنین حالاتی باید از علامت || استفاده کرد:

```
$cat file1 || cat file2 || cat file3
```

در دستور فوق، به‌محض این که اولین دستور با موفقیت انجام شود، سایر دستورها دیگر اجرا نمی‌شوند؛ به‌طور مثال، اگر cat file 1 با موفقیت اجرا نشود، سیستم به سراغ دستور بعدی می‌رود و درصورتی‌که cat file 2 موفقیت‌آمیز باشد، cat file 3 دیگر اجرا نخواه شد.

نکته: علامت | اصطلاحاً Pipe Sign نامیده می‌شود و به‌صورت تحت‌الفظی می‌توان آن را «یا» ترجمه کرد. درضمن، برای تایپ این علامت در کیبوردهای استندرد \ +shift را باید فشار داد.

۲.۳ میتود و دستورات پرکاربرد در اسکرِپت‌نویسی

۲.۳.۱ آشنایی با مفهوم I/O Redirection [۱]

اکثر سیستم‌عامل‌ها این امکان را در اختیار ما قرار می‌دهند تا ورودی (Input) از کیبورد گرفته شده و خروجی (Output) در ترمینال نمایش داده شود؛ به‌هرحال، در اسکرِپت‌نویسی Shell می‌توان خروجی را در یک فایل ذخیره کرد که به چنین کاری اصطلاحاً Output Redirection گفته می‌شود (Palmer,۲۰۰۷). علامت > برای ذخیره‌سازی خروجی در یک فایل مورد استفاده قرار می‌گیرد؛ برای مثال، دستور زیر خروجی دستور free را به فایلی به نام tmp/free.out/ می‌فرستد:

```
$ free > /tmp/free.out
```

نکته: دستور free در لینوکس میزان حافظه و فضای swap استفاده‌شده و باقی‌مانده را نشان می‌دهد.

همان‌طور که خروجی را می‌توان در یک فایل ذخیره کرد، ورودی (Input) یک دستور را نیز می‌توان یک فایل در نظر گرفت؛ پروسهٔ خواندن ورودی از یک فایل اصطلاحاً Input Redirection نامیده می‌شود که برای این‌کار از علامت > استفاده می‌شود؛ به‌طور مثال، اگر فایلی تحت عنوان script.sh با محتویات زیر داشته باشیم:

```
#!/bin/bash
echo “Line count”
wc -l < /temp/free.out
```

و دستور chmod +x script.sh را به‌منظور قابل اجراکردن آن انجام داده و سپس این فایل را با دستور script.sh/. اجرا کنیم، این دستور تعداد خطوط فایل tmp/free.out/ را شمرده و نتایج را نشان خواهد داد:

---

[۱] - input/output redirection

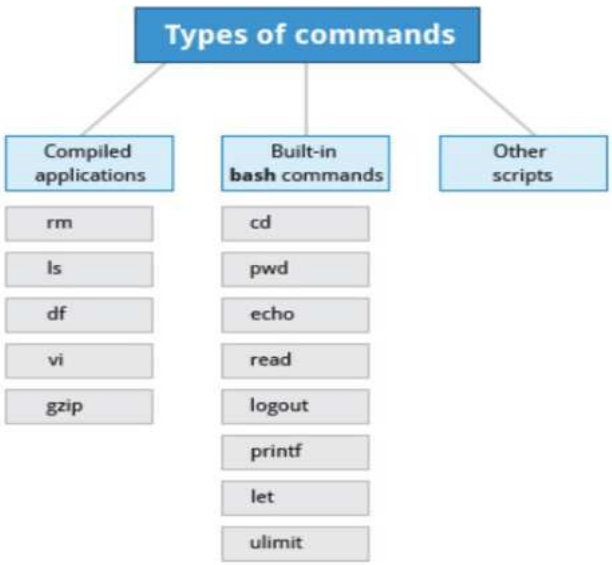
```
$./script.sh
"Line count"
3
```

۲.۳.۲ دستورهای از پیش تعریف‌شده[1] Shell

اسکرِپت‌های شل به‌منظور اجرای تعدادی دستور مورد استفاده قرار می‌گیرند که این دستورها را می‌توان به دسته‌های زیر تقسیم‌بندی کرد:

- ابزارهای کامپایل‌شده؛
- دستورهای از پیش تعریف شده Bash؛
- دیگر اسکرِپت‌ها.

ابزارهای کامپایل‌شده عبارتند از فایل‌های قابل‌اجرای باینری که می‌توان در فایل‌سیستم لینوکس به آنها دسترسی پیدا کرد که از آن جمله می‌توان به ابزارهایی چون vi،df،ls،rm و gzip اشاره کرد. لازم به تذکر است که اسکرِپت‌های شل همیشه به این ابزارها دسترسی مستقیم دارند:



شکل(۲-۳) انواع دستورها

خود Bash هم دارای دستورهای زیادِ از پیش تعریف‌شده (Built-In) است که فقط برای نمایش (خروجی) در ترمینال مورد استفاده قرار می‌گیرند (گاهی این دستورها نامی مشابه برنامه‌های قابل اجرا روی سیستم دارند که از آن جمله می‌توان به echo اشاره کرد). دستورهای از پیش تعریف‌شدهٔ Bash عبارتند از cd،pwd،echo،read،logout،printf،let و ulimit.

[1] - built-in commands

نکته: لست کامل دستورهای Built-In بش در صفحهٔ man bash قابل دسترسی است ویا با تایپ‌کردن help در ترمینال می‌توان آنها را مشاهده کرد.

۲.۳.۳ تعیین پارامترها در اسکرِپت

در بسیاری از مواقع ضرورت به استفاده از پارامتر برای یک اسکرِپت می‌باشد، پارامترها در اصل ورودی‌های اسکرِپت است که توسط کاربر در وقت نوشتن دستور مشخص می‌شود. که از جمله این پارامترها می‌توان به نام فایل، تاریخ و ... اشاره کرد. پارامترها از هر نوعی می‌توانند باشند؛ مثلا: نوع این پارامترهای ورودی هم می‌توانند عددی باشند ویا یک فایل باشد:

```
$./script.sh /tmp
$./script.sh 100 200
```

در اسکرِپت اول فایلی تحت عنوان tmp به‌عنوان پارامتر ورودی درنظر گرفته شده است و در اسکرِپت دوم دو عدد مختلف به‌عنوان پارامترهای ورودی استفاده شده است. در اسکرِپت‌نویسی شل، پارامتر با استفاده از یک علامت $ و یک عدد مشخص می‌شود؛ جدول زیر بعضی از این پارامترها را نشان می‌دهد (Palmer,۲۰۰۷):

جدول(۲-۲) انواع پارامترها

پارامتر	کاربرد
$0	نام اسکرِپت
$1	اولین پارامتر
$2، $3 و غیره	دومین، سومین و ... پارامتر
$*	تمامی پارامترها
$#	تعداد پارامترها

برای روشن‌تر شدن این مسأله، مثالی می‌زنیم؛ در ادیتور دلخواه خود فایلی تحت عنوان script.sh ساخته و محتویات زیر را داخل آن کاپی کنید:

```
#!/bin/bash
echo “The name of this program is: $0”
echo “The first argument passed from the command line is: $1”
echo “The second argument passed from the command line is: $2”
echo “The third argument passed from the command line is: $3”
echo “All of the arguments passed from the command line are: $*”
```

سپس با استفاده از دستور chmod +x به این فایل قابلیت اجرا بدهید و آن را به یک فایل اجرایی تبدیل کنید؛ سپس با درنظرگرفتن ۳ پارامتر one two three، فایل script.sh را اجرا کنید، خروجی این اسکرِپت به‌صورت زیر خواهد بود:

```
The name of this program is:./script.sh
The first argument passed from the command line is: one
The second argument passed from the command line is: two
The third argument passed from the command line is: three
All of the arguments passed from the command line are: one two three
```

تشریحات اسکرِپت فوق:

- $0 نام اسکرِپت را چاپ می‌کند که برابر است با script.sh؛
- $1 نام اولین پارامتر را چاپ می‌کند که برابر است با one؛
- $2 نام دومین پارامتر را چاپ می‌کند که برابر است با two؛
- $3 نام سومین پارامتر را چاپ می‌کند که برابر است با three؛
- $* هم نام هر سه پارامتر ورودی را چاپ می‌کند.

۲.۳.۴ جایگزینی دستورها در اسکرِپت‌نویسی

گاهی شما نیاز دارید تا نتیجهٔ یک دستور را به‌عنوان قسمتی از دستور دیگر جایگزین کنید که به این کار اصطلاحاً Command Substitution گفته می‌شود که با استفاده از دو روش مختلف انجام می‌شود:

- با قرار دادن علامت ` که معمولاً زیر دکمهٔ Esc می‌باشد، در دو طرف دستور داخلی؛
- با قرار دادن $( ) دور دستور داخلی.

هیچ فرقی نمی‌کند که از کدام روش استفاده کنیم، داخلی‌ترین دستور در یک محیط Shell جدید اجرا شده و نتیجهٔ اجرای آن در اسکرِپت اصلی در نظر گرفته می‌شود. در حقیقت، هر دستوری را می‌توان به این روش اجرا کرد و هر دو روش هم امکان جایگزینی دستورها را برایمان فراهم می‌کنند؛ مثال:

```
$ cd /lib/modules/$(uname -r)/
```

در دستور فوق خروجی دستور uname -r به‌عنوان ورودی دستور cd قرار می‌گیرد، uname –r دستور داخلی و cd دستور اصلی است.

۲.۳.۵ متغیرها در اسکرِیپت

متغیرها در برنامه‌نویسی کاربردهای زیادی دارد و شما قبلاً در مضامین برنامه‌نویسی با آنها آشنا شده‌اید و ما در اینجا به نحوهٔ استفاده از آن در اسکرِیپت‌نویسی اشاره می‌کنیم. تمامی اسکرِیپت‌ها از Variable (متغیر) که حاوی یک Value (قیمت) باشد استفاده می‌کنند، که در هر کجای دیگر اسکرِیپت می‌توانید قیمت‌های آنها را مورد استفاده قرار دهید. متغیرها از نظر تعریف دو نوع هستند، یا متغیر سیستمی هستند که از قبل در سیستم‌عامل تعریف شده می‌باشد ویا متغیر کاربر است که توسط کاربر تعریف می‌شود.

متغیرهای محیطی (Environment Variable) یکی از انواع متغیرهای سیستمی است که از آنها در اسکرِیپت‌نویسی می‌توان استفاده کرد؛ به‌عنوان مثال: PATH،HOME و HOST نمونه‌هایی از این متغیرها می‌باشند؛ وقتی که این متغیرهای محیطی را می‌خواهیم استفاده کنیم، حتماً یک علامت $ - مثل $HOME- باید قبل از آنها قرار دهیم. برای مشاهدهٔ قیمت‌های فعلی آنها نیز می‌توان به‌طور نمونه از دستور echo استفاده کرد (Palmer,۲۰۰۸) :

```
$ echo $PATH
```

درعین‌حال برای قیمت‌دهی و به‌روزرسانی قیمت‌های این نوع متغیرها ضرورت به استفاده از علامه $ نیست؛ به‌طور مثال، برای ایجاد یک متغیر تحت عنوان MYCOLOR و اختصاص رنگ آبی به آن، به‌صورت زیر عمل می‌کنیم:

```
$ MYCOLOR=blue
```

نکته: برای دستیابی به لستی از متغیرهای محیطی، می‌توان از دستورهای env و printenv استفاده کرد.

همان‌طور که در دستور فوق مشاهده می‌شود، در بین MYCOLOR و علامت = و blue هیچ فاصل‌های وجود ندارد و درصورتی‌که به اشتباه فاصل‌هایی در بین آنها قرار گیرد سیستم error خواهد داد.

۲.۳.۵.۱ خروجی گرفتن از متغیرها

به‌صورت پیش‌فرض، متغیرهایی که داخل یک اسکرِیپت ساخته می‌شوند فقط در همان اسکرِیپت در دسترس می‌باشند که کلیه فرایندهای جانبی که تحت عنوان Sub-Shell شناخته می‌شوند به چنین متغیرهایی دسترسی ندارند. برای آن که بتوانیم از طریق اسکرِیپت‌های زیرشاخه به این متغیرهای دسترسی داشته باشیم، متغیرهای ایجادشده می‌بایست بااستفاده از دستور export به Environment Variableها ارتقا یابند؛ در همین راستا داریم:

```
export VAR=value
```

توجه داشته باشیم که متغیرهای export شده به اشتراک گذاشته نمی‌شوند بلکه صرفاً کاپی می‌شوند و به همین دلیل هم هست که اگر تغییری در قیمت متغیرها توسط اسکرِپت‌های زیرشاخه صورت گیرد، این تغییرات توسط اسکرِپت اصلی مشاهده نخواهند شد. برای روشن‌تر شدن این مسأله، مثالی می‌زنیم:

```
$ SITE=AfghanAcademy
```

در این مرحله توانسته‌ایم متغیری تحت عنوان SITE ایجاد کنیم؛ حال اقدام به استفاده از دستوری تحت عنوان export می‌کنیم:

```
$ export SITE
```

اکنون دستور export را وارد کرده و انتر را بزنید، در لست متغیرهایی که نمایش داده می‌شود خواهید دید که متغیر SITE اضافه شده است.

۲.۴ آشنایی با توابع (Function)

تابع (Function) مجموعه‌یی از دستورها و دستورات shell است که تحت یک نام ذخیره می‌شوند و وظیفهٔ مشخصی را انجام می‌دهند. توابع معمولاً تعدادی پارامتر ورودی و یک خروجی دارند. برای ساخت یک تابع در اسکرِپت باید با دو موضوع آشنایی داشت:

- آشنایی با نحوهٔ ایجاد تابع؛
- آشنایی با نحوهٔ صدازدن[1] یا استفادهٔ تابع.

برای نوشتن یک تابع، نیاز به یک نام است تا بعداً به‌سادگی بتوانیم آن را صدا بزنیم (در اسکرِپت‌نویسی شل، به‌جای صدازدن تابع از اصطلاح Callکردن استفاده می‌شود)؛ نحوهٔ صحیح ساخت یک تابع به‌صورت زیر است:

```
function_name(){
 command…
}
```

برای مثال، در ادامه تابعی ساخت‌هایم تحت عنوان display که وظیفهٔ نمایش یک پیغام را دارد:

---

[1] - call

```
display () {
 echo “This is a sample function”
}
```

یک تابع می‌تواند شامل چندین خط کُد باشد و مزیت اصلی تابع این است که یک بار تعریف می‌شود و به دفعات زیاد می‌توان آن را صدا زد و استفاده کرد؛ برای مثال:

```
#!/bin/bash
display(){
echo “This is the message from the function”
echo “This parameter passed from calling process is” $1
}
display “User1”
display “Admin”
```

در خط اول نوع شل مشخص شده است، از خط دوم تا پنجم، تابعی به نام display تعریف شده است. در مثال فوق، ما در این تابع از ۲ دستوری که با دستور echo شروع می‌شوند استفاده کرده‌ایم؛ در دستور اول گفته‌ایم که عبارت This is the message from the function به‌معنای «این پیامی از طرف این تابع است» چاپ شود و در دستور دوم که در خط ۴ نشان می‌دهد که عبارت This parameter passed from calling process is به‌معنای «پارامتر پاس داده‌شده از پروسهٔ فراخوانی‌کننده برابر است با» به‌علاوه قیمت متغیر $۱ نمایش داده شود.

در مباحث قبل گفتیم که هر موقع بخواهیم به اولین پارامتر ورودی تابع دسترسی پیدا کنیم، می‌بایست از $۱ استفاده کنیم و از آنجاکه در این مثال صرفاً یک پارامتر ورودی داریم، پس $۱ هم پارامتری را که بعداً قرار است برای تابع display در نظر گرفته شود، نمایش خواهد داد. در خط‌های ۷ و ۸ هم ۲ بار اقدام به اصطلاحاً Call (فراخوانی)کردن تابع display کرده‌ایم که در مورد اول از پارامتر ورودی User۱ و در مورد دوم از پارامتر ورودی Admin استفاده کرده‌ایم. حال ابتدا همان‌طور که قبلاً گفته شد، کُدهای فوق را در فایلی با نام دلخواه همچون function.sh ذخیره کرده، سپس با استفاده از دستور زیر این فایل را قابل اجرا می‌سازیم:

```
$ chmod +x function.sh
```

اکنون به‌سادگی و با استفاده از دستور ./function.sh می‌توانیم این فایل را اجرا نماییم:

```
$./function.sh
This is the message from the function.
This parameter passed from calling process is User1
This is the message from the function.
This parameter passed from calling process is Admin
```

همان‌طور که در خروجی مشاهده می‌شود، به هر تعداد که تابع display فراخوانی شده است، دستورات قرار گرفته داخل این تابع نیز اجرا می‌شوند.

۲.۴.۱ آشنایی با نحوهٔ استفاده از دستورات شرطی

دستورات شرطی در تمامی زبان‌های برنامه‌نویسی وجود دارند و اسکرِپت‌نویسی Shell هم از این قاعده مستثنی نیست؛ زمانی‌که از یک دستور شرطی در اسکرِپت خود استفاده می‌کنیم، پروسهٔ بعدی بستگی به نتیجهٔ یکی از شرایط زیر دارد:

- مقایسهٔ ۲ عدد یا استرینگ با یکدیگر؛
- مقدار بازگشتی یک دستور (همان‌طور که قبلاً اشاره شد، عدد ۰ نشان‌دهندهٔ موفقیت‌آمیز بودن دستور است و سایر اعداد هم حاکی از عدم موفقیت یک دستوراند)؛
- چک نمودن حق دسترسی (Permisstion) به فایل.

نکته: به‌طورکلی، منظور از String (استرینگ یا رشته) مجموعه‌یی از علایم، کرکترها، اعداد و... است که معمولاً بین دو علامت ‘ , “ نوشته می‌شود مثل “ ۱۲۳۴abcdefghij @!#$٪^&&”.

برای مقایسهٔ حالت‌های مختلف شرطی از دستورات شرطی استفاده می‌شود. سه نوع دستور شرطی وجود دارد:

- دستورات شرطی ساده (simple if)
- دستورات شرطی دو طرفه (two-way if)
- دستورات شرطی تو در تو (nested if)

به‌طور خلاصه، فورم یک دستور شرطی ساده به‌صورت زیر است:

```
if TEST-COMMANDS; then CONSEQUENT-COMMANDS; fi
```

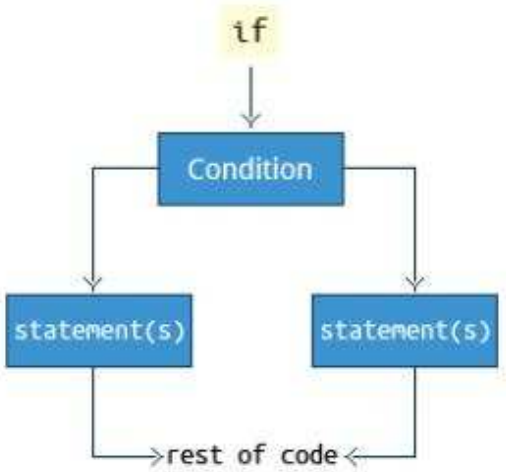
به‌عنوان مثال، دستور if زیر چک می‌کند ببیند که آیا فایلی تحت عنوان /etc/passwd روی سیستم وجود دارد یا خیر و درصورتی‌که فایل یافت شد، پیامی حاوی .etc/passwd exists/ نمایش داده می‌شود:

```
if [ -f /etc/passwd ]
then
 echo “/etc/passwd exists.”
fi
```

در اسکرِپت فوق، به نحوهٔ استفاده از [ ] خوب توجه کنید؛ به‌عبارت دیگر، شرطی را که قرار است تست کنیم، داخل علایم [ ] قرار می‌دهیم.

دستورات شرطی دوطرفه نیز به‌صورت زیر تعریف می‌شود، در دستورات شرطی دوطرفه دو حالت وجود دارد حالت صحیح و حالت غلط، دستورات حالت صحیح بین then و else نوشته می‌شوند و دستورات حالت غلط بین else و fi نوشته می‌شوند:

```
if condition
then
 statements
else
 statements
fi
```



شکل(۲-۴) فلوچارت دستورات شرطی دوطرفه

برای مثال، فایلی تحت نام ifscript.sh ایجاد نموده و اسکرِپت‌های زیر را به آن اضافه کنید:

```
#! /bin/bash
file=$1
if [ -f $file]
then
echo -e “The $file exists”
else
echo -e “The $file does not exist”
fi
```

در خط دوم پارامتر ورودی است که کاربر باید قیمت آن را وارد کند و قیمت واردشده در متغیری تحت عنوان file ذخیره می‌شود و سپس در خط سوم چک می‌شود که آیا چنین فایلی روی سیستم وجود دارد یا خیر؛ اگر فایل موجود باشد، دستور مقابل اولین echo نمایش داده می‌شود و اگر موجود نباشد، دستور مقابل دومین echo در خروجی نمایش داده می‌شود.

برای اجرای اسکرِپت، ابتدا با استفاده از دستور chmod +x ifscript.sh این فایل را قابل اجرا ساخته و درنهایت با استفاده از دستور زیر، آن را اجرا می‌کنیم:

```
$./ifscript.sh /etc/passwd
The /etc/passwd exists
```

همان‌طور که ملاحظه می‌شود، با دستور./ifscript و اختصاص دادن یک پارامتر ورودی همچون /etc/passwd این فایل را اجرا کرده و از آن‌جا که فایل /etc/passwd روی سیستم وجود دارد، دستور داخل if اجرا می‌شود و عبارت The /etc/passwd exists نمایش داده می‌شود. حال یک‌بار دیگر دستور فوق را اجرا کرده اما این‌بار مسیری که وجود خارجی روی سیستم ندارد را به‌عنوان پارامتر ورودی درنظر می‌گیریم:

```
$./ifscript.sh /etc/passwd2
The /etc/passwd2 does not exist
```

می‌بینیم باتوجه به این‌که فایلی تحت عنوان passwd۲ روی سیستم وجود ندارد، دستور داخل else اسکرِپت مدنظر، اجرا می‌گردد و پیام The /etc/ passwd۲ does not exist نمایش داده می‌شود.

۲.۴.۲ چک‌کردن فایل‌ها

به‌طورکلی، Bash تعدادی دستورات شرطی مرتبط با فایل‌ها دارد که با دستورات شرطی if می‌توانند استفاده کرد. در جدول زیر مهم‌ترین این دستورات لست شده است (Blum,۲۰۰۸; Palmer,۲۰۰۷):

جدول(۲-۳) عملگرهای شرطی برای چک‌کردن فایل‌ها

Operator	کاربرد
-e	چک‌کردن وجود یک فایل
-d	چک‌کردن وجود یک دایرکتوری
-f	چک‌کردن این‌که فایل حالت معمولی دارد؛ به‌عبارت دیگر یک لینک یا دایرکتری نیست
-s	چک‌کردن این‌که حجم فایل برابر با ۰ نیست
-g	چک‌کردن این‌که sgid فایل ست شده است
-u	چک‌کردن این‌که suid فایل ست شده است
-f	چک‌کردن این‌که فایل به اصطلاح Readable است
-w	چک‌کردن این‌که فایل به اصطلاح Writable است
-x	چک‌کردن این‌که فایل قابل اجرا است

برای روشن‌تر شدن کاربرد شرط‌های فوق‌الذکر، فایلی می‌سازیم تحت عنوان isexescript.sh و اسکرِپت‌های زیر را داخل آن کاپی می‌کنیم:

```
#! /bin/bash
file=$1
if [ -x $file ]
then
 echo “The $file is executable”
fi
```

تنها تفاوت این اسکرِپت با اسکرِپت قبلی از لحاظ ساختار این است که از شرط x- استفاده کرده‌ایم و اگر به جدول فوق توجه کنیم، می‌بینیم که این شرط مسئول چک‌کردن این مسأله است که آیا فایلی که به‌عنوان پارامتر ورودی در نظر گرفته می‌شود، قابلیت اجرا دارد یا خیر؟ برای تست‌کردن کارکرد این اسکرِپت، قبل از هر چیز ابتدا خود این فایل را قابل اجرا کرده سپس فایلی تحت عنوان tmp.sh می‌سازیم اما آن را قابل اجرا نمی‌کنیم؛ حال دستور زیر را در ترمینال تایپ کرده و انتر می‌کنیم:

```
$./isexescript.sh tmp.sh
```

می‌بینیم که هیچ‌چیزی نمایش داده نمی‌شود؛ حال با استفاده از دستور chmod +x tmp.sh این فایل را قابل اجرا کرده سپس دستور فوق را مجدداً در ترمینال وارد کرده و انتر می‌کنیم:

```
$./isexescript.sh tmp.sh
The tmp.sh is executable
```

می‌بینیم که این‌بار دستور داخل if اجرا شد و این درصورتی است که می‌توانیم با درنظرگرفتن یک دستور else، به‌سادگی برای شرایطی که فایلی درنظرگرفته‌شده به‌عنوان پارامتر ورودی قابل اجرا نیست، پیامی خاص را نمایش می‌دهد.

دستورات شرطی تو در تو نیز برای حالت‌هایی استفاده می‌شود که بیشتر از دو یا چند شرط را بخواهیم چک کنیم و ساختار دستورات آن به‌صورت زیر است:

```
if [ condition-1 ] then
        statement-1
elif [ condition-2 ]then
        statement-2
elif [ condition-3 ]then
        statement-3
elif [ condition-4 ]then
        statement-4
elif [ condition-5 ]then
        statement-5
.
.
.
else
        statement-n
fi
```

نحوهٔ اجرای این دستورات طوری است که ابتدا شرط اول چک می‌شود و اگر صحیح باشد دستورات آن اجرا شده و خاتمه می‌یابد و اگر صحیح نباشد شرط دوم چک می‌شود و به همین ترتیب ادامه می‌یابد تا به اولین شرطی صحیح برسد. بعد از رسیدن به اولین شرط صحیح دستورات آن اجرا شده و باقی شرطها بررسی نمی‌شود و خاتمه می‌یابد (Blum,۲۰۰۸).

۲.۴.۳ کاربرد دستورات شرطی if در کنار استرینگ‌ها (String)

علاوه‌بر تست‌کردن فایل‌ها، از دستورات if برای چک‌کردن استرینگ‌ها نیز می‌توان استفاده کرد و برای چنین مواردی از ساختار زیر استفاده می‌کنیم:

```
if [ string1 == string2 ] ; then
 ACTION
fi
```

برای فهم کاربرد دستورات شرطی تو در تو و استرینگ‌ها، به مثالی که در ادامه آمده است توجه کنید. فایلی می‌سازیم تحت عنوان stringscript.sh و آن را همان‌طور که قبلاً توضیح داده شد، قابل اجرا می‌سازیم؛ سپس اسکرِپت زیر را داخل آن کاپی می‌کنیم:

```
#! /bin/bash
# Section that reads input
echo “Enter any color code (R or Y or G):”
read COLOR
# Section that comapres the entry and displays a message
if [ “$COLOR” == “R” ]
then
echo “Stop! Leave the way for others”
elif [ “$COLOR” == “Y” ]
then
echo “GET ready; Your way will be open shortly”
elif [ “$COLOR” == “G” ]
then
echo “Go Ahead. It is your turn to go”
else
echo “Incorrect color code”
fi
```

حال اقدام به تست اسکرِپت فوق می‌کنیم:

```
$./stringscript.sh
Enter any color code (R or Y or G):
```

می‌بینیم که اصطلاحاً یک Prompt (سوال) نمایش داده می‌شود، که باید در پاسخ به آن، یکی از حروف

R،Y ویا G که نمایندهٔ کلمات به‌ترتیب Red به‌معنای «سرخ»، Yellow به‌معنای «زرد» و Green به‌معنای «سبز» است را انتخاب کنیم؛ به‌طور مثال حرف G را تایپ کرده و انتر می‌کنیم:

```
$./stringscript.sh
Enter any color code (R or Y or G):
G
Go Ahead. It is your turn to go
```

می‌بینیم که دستورهای قرار گرفته در شرطی که چک می‌کند ببیند آیا قیمت متغیر COLOR برابر با G است یا خیر، اجرا شده و عبارت Go Ahead. It is your turn to go نمایش داده می‌شود؛ و اگر حروف R و Y را هم وارد کنیم، دستورات مرتبط به آنها نمایش داده خواهد شد. حال ببینیم اگر حرفی از پیش تعریف‌نشده همچون B را وارد کنیم چه اتفاقی می‌افتد:

```
$./stringscript.sh
Enter any color code (R or Y or G):
B
Incorrect color code
```

می‌بینیم که دستور else آخر اجرا می‌شود چراکه حرف B با هیچ‌کدام از معیارهای تعریف‌شده در این برنامه هم‌خوانی ندارد؛ به‌عنوان مثال پایانی، یک‌بار هم قصد داریم همان حرف G اما این‌بار به‌صورت کوچک (g) را وارد نماییم:

```
$./stringscript.sh
Enter any color code (R or Y or G):
g
Incorrect color code
```

می‌بینیم که مجدداً بخش مرتبط با else اجرا می‌شود درحالی‌که ممکن است فکر کنید حروف G و g هردو به یک چیز اشاره دارند.

نکته: در دنیای کمپیوتر، کلیه حروف اعم از حروف خورد و کلان به‌عنوان دو چیز مجزا و متفاوت از یکدیگر شناخته می‌شوند؛ لذا حروف G و g به‌عنوان ۲ حرف متفاوت از یکدیگر به شمار می‌روند. آنچه در اسکرِپت فوق نیاز به توضیح دارد این است که در تفسیر اسکرِپت فوق باید بگوییم که پس از اجرا، ابتدا شرط داخل اولین if چک می‌شود که تست می‌کند ببیند آیا قیمت متغیر COLOR برابر با حرف R هست یا خیر؛ اگر این مقدار برابر بود، دستورهای مرتبط با این if اجرا می‌شوند و برنامه پایان می‌یابد و در غیر این‌صورت، مفسر (Bash) به سراغ دستور elif اول می‌رود که مسئول چک‌کردن برابری قیمت متغیر COLOR با حرف Y

است. به‌همین ترتیب کلیه elifهای برنامه چک می‌شوند و درصورتی هم که هیچ کدام از آنها اجرا نشد، برنامه به‌صورت خودکار دستورهای داخل else را اجرا می‌کند.

هشدار: درصورتی‌که دستور fi که در پایان دستورات شرطی می‌آید را فراموش کنیم، یک خطا یا error به حساب می‌آید و اجرا نمی‌شود؛ لذا همواره به یاد داشته باشیم که دستورات شرطی خود را «پایان دهیم» که به‌سادگی می‌توان این‌کار را با دستور fi انجام داد.

۲.۴.۴ کاربرد دستورات شرطی if در کنار اعداد

بعض اوقات در اسکرِپت‌نویسی Shell نیاز داریم تا اعداد را در دستورات شرطی تست کنیم که برای این منظور می‌توانیم از عملگرهای زیر استفاده کنیم (Blum,۲۰۰۸):

جدول(۲-۴) عملگرهای مقایسه‌یی برای اعداد

علامت ریاضی	operator	کاربرد
=	-eq	برابر با
≠	-ne	نابرابر با
<	-gt	بزرگ‌تر از
>	-lt	کوچک‌تر از
≤	-ge	بزرگ‌تر از یا برابر با
≥	-le	کوچک‌تر از یا برابر با

ساختار دستورات شرطی برای مقایسهٔ اعداد به‌صورت زیر است:

```
exp1 -op exp2
```

برای درک بهتر، برنامه‌یی ساده می‌نویسیم که در آن سن کاربر را گرفته سپس پیامی را نمایش می‌دهیم که آیا وی در دههٔ سوم زندگی‌اش، دههٔ چهارم یا دههٔ پنجم زندگی‌اش به‌سر می‌برد ویا سن وارد شده توسط وی خارج از دامنهٔ درنظرگرفته‌شده در برنامه است. برای این‌منظور، فایلی تحت عنوان agescript.sh در ادیتور دلخواه خود ایجاد کرده و اسکرِپت‌های زیر را داخل آن وارد می‌کنیم:

```
#! /bin/bash
# prompt for a use age
echo “Please enter your age:”
read AGE
if [ “$AGE” -lt 20 ]|| [ “$AGE” -ge 50 ]
then
echo “Sorry, you are out of the age range.”
elif [ “$AGE” -ge 20 ] && [ “$AGE” -lt 30 ]
then
echo “You are in your 20s”
elif [ “$AGE” -ge 30 ] && [ “$AGE” -lt 40 ]
then
echo “You are in your 30s”
elif [ “$AGE” -ge 40 ] && [ “$AGE” -lt 50 ]
then
echo “You are in your 40s”
fi
```

ابتدا برنامهٔ فوق را اجرا کرده و نحوهٔ عملکرد آن را مشاهده می‌کنیم سپس به تفسیر کُد منبع (Source code) آن می‌پردازیم؛ برای این منظور، همان‌طور که قبلاً توضیح داده شد، ابتدا دستور chmod +x agescript.sh را اجرا می‌کنیم تا این فایل به‌اصطلاح Executabe (قابل‌اجرا) شود؛ سپس دستور ./agescript.sh را در خط فرمان (CLI) وارد کرده و برای مثال عدد ۷۵ را وارد می‌کنیم:

```
$./agescript.sh
Please enter your age:
75
Sorry, you are out of the age range.
```

می‌بینیم باتوجه به این‌که سن واردشده بیش از ۵۰ است، این پیام صادر شده است. حال مجدداً برنامه را اجرا کرده و این‌بار عدد ۲۲ را وارد می‌کنیم:

```
$./agescript.sh
Please enter your age:
22
You are in your 20s
```

سپس یک‌بار دیگر برنامه را اجرا کرده و این‌بار عدد ۳۳ را وارد می‌کنیم:

```
$./agescript.sh
Please enter your age:
33
You are in your 30s
```

درنهایت، یک‌بار دیگر برنامه را اجرا کرده و این‌بار عدد ۴۷ را وارد می‌کنیم:

```
$./agescript.sh
Please enter your age:
47
You are in your 40s
```

می‌بینیم که این برنامه به‌درستی کار می‌کند اما ساختاری در این برنامه مورد استفاده قرار گرفته که به‌نوعی جدید است و آن‌هم چیزی نیست جز علایم || و &&؛ به‌طورکلی در اکثر زبان‌های برنامه‌نویسی و همچنین Bash، دستور || به‌معنای «یا» و دستور && به‌معنای «و» است. جدول زیر شامل برخی از مهم‌ترین عملگرهای منطقی است که در اسکرِپت‌نویسی مورد استفاده قرار می‌گیرند (Blum,۲۰۰۸):

جدول(۲-۵) عملگرهای منطقی

Operator	عملکرد	توضیحات
&&	و	درصورتی‌که هر دو شرط برابر با True باشند، اسکرِپت ادامه می‌یابد.
||	یا	درصورتی‌که هریک از شرط‌ها برابر با True باشد، اسکرِپت ادامه می‌یابد.
!	مخالف	اسکرِپت فقط در صورتی ادامه می‌یابد که شرط برابر با False گردد.

به یاد داشته باشید زمان‌هایی که از چندین عملگر منطقی با استفاده از عملگر && (به معنای و) در کنار یکدیگر استفاده می‌کنید، به محض این‌که شرط برابر با False (مقدار 0 یا نادرست) گردد، برنامه متوقف

می‌گردد؛ برای مثال، اگر شرط‌های A && B && C را داشته باشیم و مقدار A برابر با True باشد اما B برابر با False، شرط C هرگز چک نخواهد شد.

همچنین در صورت استفاده از عملگر || (به معنای یا) برنامه به محض این که یکی از شرط‌ها برابر با True شود متوقف خواهد شد. به عبارت دیگر، اگر شرط‌های A || B || C را داشته باشیم و مقدار A برابر با False باشد اما B برابر با True باشد، شرط C هیچ وقت چک نمی‌شود.

در تفسیر اسکرِپت فوق باید بگوییم که در دستور if این شرط را گذاشته‌ایم که اگر سن کاربر کوچک‌تر از (-lt) ویا بزرگ‌تر از یا برابر با (-ge) بود، عبارت Sorry, you are out of the age range. نمایش داده شود. درواقع، هر زمانی‌که از || استفاده می‌کنیم، تازمانی‌که یکی از شرط‌ها برآورده شود، دستور مرتبط اجرا خواهد شد اما در مورد && این‌گونه است که حتماً هردو شرط می‌بایست برآورده شوند.

به‌همین ترتیب در کلیه دستورات elif این برنامه هم از دستور && استفاده کرده و همان‌طور که در بالا گفته شد، در این موارد حتماً پاسخ به هردو شرط می‌بایست مثبت باشد تا دستور مربوطه اجرا گردد.

۲.۴.۵ محاسبات ریاضی در اسکرِپت‌نویسی

به‌منظور انجام محاسبات پای‌های ریاضیاتی در شل، سه روش مختلف وجود دارد:

- استفاده از دستور expr؛
- استفاده از $((...))؛
- استفاده از دستور let.

برای مثال، به‌منظور جمع‌کردن اعداد ۷ و ۳ با یکدیگر از ساختار زیر استفاده می‌کنیم:

```
$ expr 7 + 3
10
```

همین نتیجه را با راهکار دیگری به‌صورت زیر نیز می‌توان به دست آورد:

```
$ echo $(expr 7 + 3)
10
```

کاربرد دستور let هم به‌صورت زیر است:

```
$ let x=( 1 + 2); echo $x
3
```

اکنون عدد ۷ را می‌توان به شکلی دیگر با عدد ۳ جمع کرد:

```
$ echo $((x+7))
10
```

۲.۴.۶ کار با String

به‌طورکلی، منظور از String (استرینگ یا رشته) مجموعه‌یی از کرکترهایی است که می‌توانند شامل حروف، اعداد، علایم خاص و ... باشند؛ به‌طور مثال، مواردی نظیر abcde 123،123،abcde، abcde-123 و &abcde=%123 همگی استرینگ محسوب می‌شوند. String معمولاً در بین دو علامهٔ "..." نوشته می‌شوند، به‌منظور مقایسه، مرتب سازی و همچنین یافتن طول یک استرینگ عملگرهایی در لینوکس در نظر گرفته شده که عبارتند از (Blum, 2008):

جدول(۲-۶) عملگرهای مقایسه‌یی برای رشته‌ها

Operator	استفاده
[[string1 > string2]]	ترتیب ۲ استرینگ string1 و string 2 را با یکدیگر مقایسه می کند.
[[string1 == string2]]	کرکترهای string 1 را با کرکترهای string 2 مقایسه می کند.
myLen1=${#string1}	طول string 1 را در متغیری تحت عنوان myLen 1 ذخیره می‌سازد.

به‌منظور روشن‌تر شدن نحوهٔ به کارگیری عملگرهای فوق، فایلی می‌سازیم تحت عنوان string.sh و کُدهای زیر را داخل آن می‌نویسیم:

```
#! /bin/bash
if [ $1 == $2 ]
then
echo "The first string, $1, is the same as the second string, $2."
else
echo "The first string, $1, is not the same as the second string, $2."
fi
```

همان‌طور که در کُد فوق مشخص است، در خط اول مسیر مفسر «بش» را داده سپس در خط دوم از یک دستور شرطی (if) استفاده کرده‌ایم بدین شکل که ۲ پارامتر ورودی تحت عناوین $1 و $2 را گرفته و آنها را با یکدیگر مقایسه می‌کند. اگر قیمت‌های ورودی یکسان باشند، دستور قرار گرفته پس از کلمهٔ کلیدی then نمایش داده می‌شود و در غیر این صورت، دستور قرار گرفته پس از else و درنهایت هم با دستور fi این

اسکرِپت به پایان می‌رسد. پیش از هر چیز، با استفاده از دستور chmod +x string.sh می‌بایست این برنامه را قابل اجرا سازیم؛ سپس یک‌بار کلمات Afghan و Academy را به‌عنوان پارامترهای ورودی در نظر می‌گیریم:

```
bash string.sh Afghan Academy
The first string, Afghan, is not the same as the second string, Academy.
```

طوری که دیده می‌شود، دستور داخل else اجرا می‌شود. حال یک بار دیگر این برنامه را اجرا کرده و قیمت‌ها یکسانی را به‌عنوان پارامترهای اول و دوم در نظر می‌گیریم:

```
bash string.sh Afghan Afghan
The first string, Afghan, is the same as the second string, Afghan.
```

می‌بینیم که به‌درستی دستور قرار گرفته پس از then اجرا شد. حال می‌خواهیم برنامهٔ کوچک دیگری را بنویسیم که وجود یا عدم وجود یک فایل را تست کند؛ برای این منظور، فایلی تحت عنوان file.sh ساخته و کُدهای زیر را داخل آن می‌نویسیم:

```
#! /bin/bash
FILE=$1
if [ -f $FILE ]
then
 echo “File $FILE exists.”
else
 echo “File $FILE does not exist.”
fi
```

همان‌طور که مشخص است، ابتدا ورودی را گرفت و آن را داخل متغیری تحت عنوان FILE ذخیره می‌کنیم سپس چک می‌کنیم تا ببینیم در دایرکتری قرار گرفته در آن، چنین فایلی وجود دارد یا خیر؛ به‌طور مثال: از طریق دستور زیر تست می‌کنیم ببینیم فایلی که پیش از این تحت عنوان string.sh ساختیم وجود دارد یا خیر:

```
bash file.sh string.sh
File string.sh exists.
```

حال اگر فایلی که وجود خارجی نداشته باشد - مثلاً string2.sh- را به‌عنوان پارامتر ورودی این برنامه در نظر بگیریم، مسلماً دستور داخل else اجرا خواهد شد.

۲.۴.۶.۱ دسترسی به بخشی از یک استرینگ

گاهی به‌جای مقایسهٔ کل یک استرینگ، ما نیاز داریم تا صرفاً بخشی از آن را با چیز دیگری مقایسه کنیم؛ به‌طور مثال، برای به‌دست‌آوردن اولین کرکتر یک استرینگ، می‌بایست از ساختار ${string:0:1} استفاده نماییم که در آن ۰ که اصطلاحاً Offset نامیده می‌شود و جایی است که تمایل داریم شمارش از آنجا شروع شود و ۱ تعداد کرکترهایی است که قصد داریم از نقطهٔ Offset به بعد استخراج شوند. برای روشن‌تر شدن این مسأله، مثالی می‌زنیم. با استفاده از دستور export، یک متغیر محیطی، تحت هر عنوانی که تمایل داشته باشیم، می‌سازیم:

```
export fullname="User1.Admin"
```

با استفاده از دستور export و انتخاب نامی دلخواه همچون fullname و درنظرگرفتن مقداری برای آن همچون User۱.Admin به‌سادگی توانستیم یک Environment Variable ایجاد کنیم. حال در ادامه قصد داریم بخش اول این نام - یعنی User۱- را جدا کنیم:

```
firstname=${fullname:0:6}
```

می‌بینیم که متغیر جدیدی تحت عنوان firstname ساخته و با درنظرگرفتن آفست ۰ سپس عدد ۶، دستور داده‌ایم که شروع شمارش کرکترها از ابتدای استرینگ یا نقطهٔ ۰ بوده و تا ۶ کرکتر بعد را در بر گیرد. برای نمایش‌دادن خروجی هم از دستور زیر استفاده می‌کنیم:

```
echo $firstname
User1
```

اگر هم بخواهیم در یک استرینگ، کلیه کرکترهای قرار گرفته پس از یک Dot (دات یا نقطه) را جدا کنیم، می‌توانیم از دستور زیر استفاده کنیم:

```
lastname=${fullname#*.}; echo $lastname
Admin
```

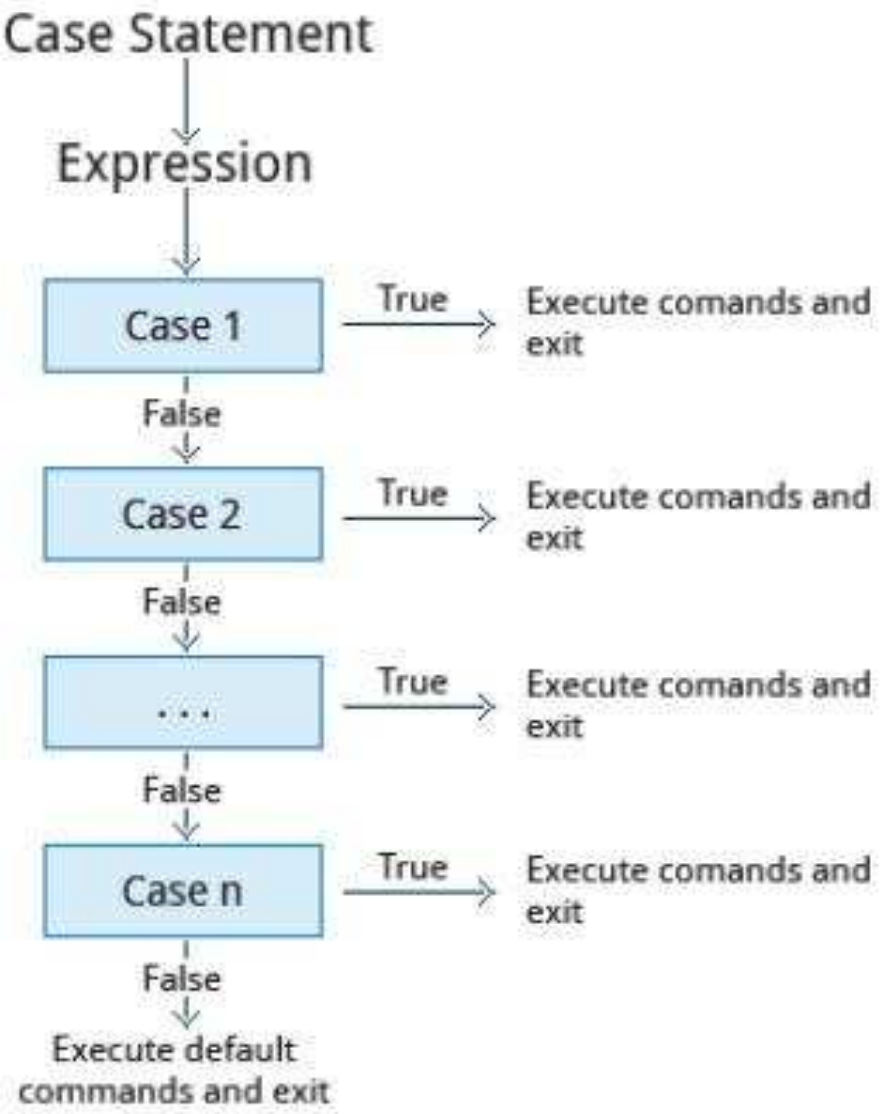
می‌بینیم که بخشی از قیمت متغیر fullname که پس از نقطه قرار گرفته بود -Admin- چاپ شد.

۲.۴.۷ نحوهٔ استفاده از دستور case

به‌طور کلی، این ساختار از جمله ساختارهای شرطی تو در تو به حساب می‌آید، و مشابه ساختار شرطی elif است، با این تفاوت که در این ساختار مقایسه فقط از نظر برابری انجام می‌شود و خورد و کلانی و... مقایسه نمی‌شود و استفاده از آن مزایای زیر را دارد (Palmer,۲۰۰۷):

- خواندن و نوشتن این نوع دستورات آسان‌تر است.
- بهتر از چندین دستور if و elif است.
- این امکان را به برنامه‌نویس می‌دهد تا قیمت یک متغیر را با چندین قیمت مختلف مقایسه نماید.

- این ساختار مغلق بودن اسکرِپت را کاهش می‌دهد.



شکل(۲-۵) فلوچارت ساختار case

همان‌طور که مشاهده می‌شود، ساختار یک دستور case در عکس فوق به تصویر کشیده شده است. اگر بخواهیم این ساختار را در قالب کُد بررسی کنیم، می‌بایست کُدهای زیر را در نظر بگیریم:

```
case $variable in
 pattern-1)
 commands
 ;;
 pattern-2)
 commands
 ;;
 pattern-3| pattern-4| pattern-5)
 commands
 ;;
 pattern-N)
 commands
 ;;
*)
 commands
 ;;
esac
```

در تفسیر کُدهای فوق بایست گفت که متغیر $variable ورودی دستور case است و pattern-1،pattern-2 الی آخر هم به منزلهٔ قیمت‌هایی هستند که قرار است متغیر ما با آنها مقایسه شود. همچنین در پایان هر دستور می‌بایست از علامت ;; استفاده کرد.

هشدار: لازم به تذکر است که ۲ علامت ; می‌بایست بدون فاصله و به‌صورت چسبیده به هم تایپ شوند که در غیر این صورت با error رو به رو خواهیم شد.

دستور (* به‌عنوان دستور پیش‌فرض عمل می‌کند و اگر هیچ گزینه‌یی مطابقت نکند، این گزینه اجرا خواهد شد. دستور esac هم به منزلهٔ پایان دستور case است.

برای روشن‌تر شدن نحوهٔ کاربرد دستور case، فایلی ایجاد می‌کنیم تحت عنوان case.sh و کُدهای زیر را داخل آن می‌نویسیم:

```
#! /bin/bash
echo “Please enter a letter:”
read character
case $character in
“a” | “A”)
echo “You have typed a vowel!”
;;
“e” | “E”)
echo “You have typed a vowel!”
;;
“i” | “I”)
echo “You have typed a vowel!”
;;
“o” | “O”)
echo “You have typed a vowel!”
;;
“u” | “U”)
echo “You have typed a vowel!”
;;
*)
echo “You have typed a consonant!”
;;
esac
```

همان‌طور که مشخص است، در خط اول مسیر مفسر bash را داده‌ایم سپس در خط دوم با استفاده از دستور echo از کاربر می‌خواهیم که حرفی را وارد کند و حرف واردشده را در متغیری تحت عنوان character ذخیره می‌سازیم. در ادامه، دستور case را نوشته و یک علامت & به نام متغیر character اضافه کرده سپس کلمهٔ کلیدی in را می‌نویسیم که به این معناست که قیمت متغیر $character در حالاتی که در ادامه می‌آیند می‌بایست چک شود.

به‌عنوان شرط اول، گفته‌ایم اگر حرف ورودی کاربری حرف a یا A بود، عبارت ! You have typed a vowel نمایش داده شود. به همین ترتیب، سایر حروف صدادار (o،i،e و u) را در شرط‌های بعدی آورده‌ایم و درنهایت هم از دستور (* استفاده کرده‌ایم که اگر هیچ‌کدام از موارد فوق برابر با True نشد، این شرط اجرا

گردد و پیغام !You have typed a consonant را نمایش دهد و در پایان هم با نوشتن دستور esac به این دستور case پایان می‌دهیم. همان‌طور که پیش از این آموزش داده شد، ابتدا با استفاده از دستور chmod +x case.sh این فایل را قابل اجرا کرده سپس آن را تست می‌کنیم:

```
bash case.sh
Please enter a letter:
a
You have typed a vowel!
```

برای تست اول، حرف a را تایپ کرده و انتر می‌کنیم؛ می‌بینیم که شرط اول برقرار می‌گردد. حال یک بار دیگر این برنامه را اجرا کرده این بار حرفی همچون z را وارد می‌کنیم:

```
bash case.sh
Please enter a letter:
z
You have typed a consonant!
```

می‌بینیم از آنجاکه هیچ‌کدام از شرط‌های در نظر گرفته شده دستور case برابر با True نشد، در نهایت دستور قرار گرفته پس از (*) اجرا گردید.

۲.۵ کار با حلقه‌ها (Loops)

چنانچه نیاز داشته باشیم تا دستوری را باربار تکرار کنیم، می‌بایست با مفهومی تحت عنوان Loop (یا حلقه) آشنایی داشته باشیم. معمولاً اجرای حلقه (Loop)ها تا زمانی ادامه می‌یابد که شرطی از پیش تعریف‌شده برابر با True یا False باشد. در اسکرِپت‌نویسی Shell معمولاً ۳ نوع حلقه (Loop) استفاده می‌شود که عبارتند از: for،while و until.

۲.۵.۱ آشنایی با for

for زمانی استفاده می‌شود که بخواهیم عملی را روی تک تک اقلام یا سطرهای یک لست انجام دهیم تا در نهایت کلیه اقلام یا سطرهای آن لست به پایان برسند. ساختار این نوع حلقه (Loop) به شرح زیر است:

```
for variable-name in list
do
 execute one iteration for each item in the list until the list is finished
done
```

در تفسیر کُد فوق باید بگوییم که variable-name را می‌توان با هر نام دلخواهی جایگزین کرد و این درحالی‌است که list هم می‌بایست حاوی لستی از اقلام یا سطرهای مختلف باشد (این اقلام یا سطرها

می‌توانند اعداد، استرینگ و ... باشند). برای روشن‌تر شدن نحوهٔ کاربرد حلقه (Loop) از نوع for، فایلی می‌سازیم تحت عنوان for.sh و کُدهای زیر را داخل آن می‌نویسیم:

```
#!/bin/sh
for i in 1 2 3 4 5
do
 echo “Looping number $i”
done
```

همان‌طور که مشخص است، در خط اول مسیر مفسر bash را داده‌ایم و در خط دوم ابتدا کلمهٔ کلیدی for را نوشته سپس نام دلخواهی برای متغیر خود همچون i (ابتدای حرف Index به معنای اندیس) در نظر می‌گیریم. در ادامه، می‌بایست به این حلقه (Loop) دستور دهیم تا داخل یک لست شروع به گشتن کند؛ برای همین منظور، کلمهٔ کلیدی in را نوشته سپس اعداد ۱ الی ۷ را به‌عنوان لستی از اقلام یا سطرهای مختلف در نظر می‌گیریم. در ادامه، ابتدا کلمهٔ کلیدی do را نوشته سپس دستوری را که تمایل داریم اجرا شود، می‌نویسیم که در این مثال استفاده از دستور echo برای چاپ‌کردن عبارت Looping number $i است که $i هم همان متغیری است که پیش از این تعریف کردیم و درنهایت کلمهٔ کلیدی done را می‌نویسم که به معنای خاتمهٔ حلقه است. نتیجهٔ اجرا در زیر آمده است:

```
$ bash for.sh
Looping number 1
Looping number 2
Looping number 3
Looping number 4
Looping number 5
Looping number 6
Looping number 7
```

می‌بینیم که دستور echo “Looping number $i“ به تعداد اقلام یا سطرهای لست ورودی، یعنی ۷ بار، تکرار شده و تا زمانی ادامه می‌یابد که دیگر هیچ آیتمی در لست باقی نمانده باشد.

حال در مثال بعد قصد داریم دستور شرطی if با حلقه for را ادغام نماییم. برای این منظور، فایلی تحت عنوان for_and_if.sh ساخته و کُدهای زیر را داخل آن می‌نویسیم:

```
#!/bin/sh
for i in {1..20}
do
 if [ $i == 10 ]
 then
 echo “$i is the first half of the list”
 else
 echo “Looping number $i”
 fi
done
```

در تفسیر کُدهای فوق باید بگوییم که با ساختار جدید همچون {۱..۲۰} مواجه هستیم. این ساختار که مانند مجموعه می‌ماند، دامنه‌یی از اعداد در بین ۱ تا ۲۰ را ایجاد می‌کند. علاوه بر این، در خط ۴ شرط گذاشته‌ایم که اگر قیمت متغیر $i برابر با ۱۰ بود، دستور “$i is the first half of the list“ echo نمایش داده شود و در غیر این صورت، دستور داخل else نشان داده شود. حال پس از اجرایی‌کردن این فایل، آن را تست می‌کنیم:

```
$ bash for_and_if.sh
Looping number 1
Looping number 2
Looping number 3
Looping number 4
Looping number 5
Looping number 6
Looping number 7
Looping number 8
Looping number 9
10 is the first half of the list
Looping number 11
Looping number 12
Looping number 13
Looping number 14
Looping number 15
Looping number 16
Looping number 17
Looping number 18
Looping number 19
Looping number 20
```

می‌بینیم که این برنامه به درستی کار کرده و اعداد ۱ الی ۲۰ را نمایش می‌دهد.

۲.۵.۲ آشنایی با حلقه while

while تازمانی‌که شرط برابر با True باشد، ادامه یافته و دستورات درنظرگرفته‌شده را اجرا می‌کند. ساختار این نوع حلقه (Loop) به شرح زیر است:

```
while condition is true
do
 Commands for execution
done
```

به‌عنوان condition (شرط) می‌توان هر نوع شرط یا عبارت شرطی را استفاده کنیم که معمولاً در بین دو علامت [ ] قرار می‌گیرد. علاوه بر این، دستورهایی را که تمایل داریم چندین دفعه تکرار شوند، باید در بین ۲ کلمهٔ کلیدی do و done قرار دهیم. برای روشن‌تر شدن نحوهٔ استفاده از while، فایلی تحت عنوان while.sh ساخته و کُدهای زیر را داخل آن می‌نویسیم:

```
#!/bin/bash
counter=1
while [ $counter -le 10 ]
do
 echo $counter
 ((counter++))
done
echo “All done”
```

در تفسیر کُدهای فوق باید بگوییم که در خط ۲ متغیری تحت عنوان counter ایجاد کرده و قیمت اولیهٔ ۱ را هم برای آن در نظر گرفته‌ایم، در خط ۳ این شرط را در نظر گرفته‌ایم که اگر قیمت متغیر $counter کوچک‌تر یا برابر با ۱۰ بود (-le)، این حلقه ادامه یابد. سپس در خط ۵ قیمت متغیر $counter را چاپ کرده و در خط بعد هم با استفاده از (( )) و استفاده از علایم ++ یک واحد به قیمت متغیر $counter اضافه می‌کنیم. حال پس از اجرایی‌کردن این فایل، آن را تست می‌کنیم:

```
$ bash while.sh
1
2
3
4
5
6
7
8
9
10
All done
```

می‌بینیم که این حلقه تازمانی‌که قیمت متغیر $counter کوچک‌تر یا مساوی با عدد ۱۰ است ادامه یافته اما به محض False شدن شرط در نظر گرفته شده، از حلقه خارج خواهیم شد و درنهایت هم دستور " echo All done" نمایش داده می‌شود.

۲.۵.۳ آشنایی با until

until تازمانی‌که شرط درنظرگرفته‌شده برابر با False باشد، دستورات حلقه (Loop) را اجرا می‌کند که با این بیان، نقطهٔ مقابل حلقه while است. به عبارت دیگر، به محض آنکه شرط برابر با True شود، این نوع حلقه خاتمه می‌یابد. ساختار این نوع حلقه (Loop) به شرح زیر است:

```
until condition is false
do
 Commands for execution
done
```

مشابه while، دستوراتی را که می‌خواهیم داخل این نوع حلقه (Loop) اجرا شوند باید در بین do و done قرار دهیم. علاوه بر این، هرگونه دستور یا عملگری را هم می‌توان به‌عنوان شرط برای until در نظر گرفت. به منظور روشن‌تر شدن نحوهٔ کاربرد until، فایلی تحت عنوان until.sh ساخته و کُدهای زیر را داخل آن می‌نویسیم:

```
#!/bin/bash
counter=1
until [ $counter -gt 10 ]
do
 echo $counter
((counter++))
done
echo “All done”
```

همان‌طور که ملاحظه می‌شود، ساختار بسیار شبیه به حلقه از نوع while است اما به‌عنوان شرط until، از عملگر gt- به معنای «بزرگ‌تر از» استفاده کرده‌ایم. قبلاً گفتیم که این نوع حلقه (Loop) تا زمانی ادامه می‌یابد که شرط برابر با False باشد و به محض True شدن آن، از حلقه (Loop) خارج خواهیم شد. حال یک بار این برنامه را پس از اجرایی‌کردن فایلش، تست می‌کنیم:

```
bash until.sh
1
2
3
4
5
6
7
8
9
10
All done
```

می‌بینیم که خروجی دقیقاً شبیه به خروجی فایل while.sh است با این تفاوت که در این برنامه شرط گذاشته‌ایم به محض این‌که قیمت متغیر $counter بزرگتر از عدد ۱۰ شد، از حلقه (Loop) خارج شده و ادامهٔ برنامه اجرا گردد.

۲.۶ پروسهٔ عیب‌یابی (debugging)

در اسکرِپت‌نویسی هم مانند کُدنویسی در هر زبان برنامه‌نویسی دیگری ممکن است با تعدادی خطا (Bug) مواجه شویم. این خطا (Bug)ها ممکن است به دلیل وجود error در اسکرِپت همچون خطای ساختاری، عدم وجود یک فایل مورد نیاز ویا عدم داشتن صلاحیت‌های مورد نیاز برای عملی ساختن یک دستور ایجاد شوند.

بسیاری از مدیرهای حرف‌های سیستم‌های لینوکس علاوه بر آشنایی با نحوهٔ اسکرِپت‌نویسی، با استراتیژی‌های عیب‌یابی (Debug)کردن اسکرِپت‌ها نیز آشنا هستند طوری که از دید برخی برنامه‌نویسان، تسلط داشتن به مهارت عیب‌یابی (Debugging) به مراتب مهم‌تر از مهارت اسکرِپت‌نویسی است چرا که اسکرِپت‌نویسی را هر کسی آشنا است اما عیب‌یابی (Debug)کردن کاری است که از عهدهٔ هر شخصی برنمی‌آید!

به‌طور کلی، پیش از اقدام‌کردن برای رفع errorها، باید بدانیم که منبع error کجا است. در اسکرِپت‌نویسی Bash، شما می‌توانید با اجرای دستور bash –x./script_file حالت (Debug Mode) را فعال نمایید و این درحالی‌است که فعال‌سازی این وضعیت به دلایل زیر کمک می‌کند تا بتوانید ریشهٔ مشکل را بیابید:

- هر دستوری قبل از اجرا نمایش داده می‌شود.
- قابلیت عیب‌یابی (Debug)کردن صرفاً بخشی از اسکرِپت با استفاده از دستورات x- و x+ وجود دارد.

برای روشن‌تر شدن نحوهٔ عیب‌یابی (Debug)کردن یک اسکرِپت در بخش، فایلی تحت عنوان debug.sh ساخته و کُدهای زیر را داخل آن می‌نویسیم:

```
#! /bin/bash
echo “Enter your title and fullname”
read name
echo “My title is ${name:0:2}”
echo “My fullname is ${name#*.}”
```

با توجه به مباحث قبلی، به نظر می‌رسد که دیگر نیازی به توضیح این که کُدهای فوق چه کاری انجام می‌دهند نباشد. برای اجرای این اسکرِپت، ابتدا با استفاده از دستور chmod +x debug.sh آن را اجرایی کرده سپس یک بار آن را تست می‌کنیم:

```
$ bash debug.sh
Enter your title and fullname
Mr. User1 Admin
My title is Mr
My fullname is User1 Admin
```

می‌بینیم که از Offset صفر تا کرکتر دوم متغیر name برای نمایش لقب (Mr) استفاده شده و در خط بعد هم دستور داده‌ایم که هر آنچه پس از نقطه قرار گرفته به‌عنوان fullname (نام و تخلص) در نظر گرفته شود. حال قصد داریم ببینیم که به چه شکل می‌توانیم این اسکرِپت را برای عیب‌یابی (Debug)کردن آماده سازیم؛ برای این منظور، کُدهای فوق را به‌صورت زیر تغییر می‌دهیم:

```
#! /bin/bash
echo “Enter your title and fullname”
set -x
read name
echo “My title is ${name:0:2}”
echo “My fullname is ${name#*.}”
set +x
```

پیش از تشریح کُدهای اضافه‌شدهٔ فوق، ابتدا اسکرِپت را یک بار اجرا می‌کنیم:

```
$ bash debug.sh
Enter your title and fullname
+ read name
Mr. User1 Admin
+ echo 'My title is Mr'
My title is Mr
+ echo 'My fullname is User1 Admin'
My fullname is User1 Admin
+ set +x
```

همان طور که ملاحظه می‌شود، پس از نوشتن دستور set -x، خط به خط کُدهای نوشته‌شده در اسکرِپت که پس از دستور set -x قرار گرفته‌اند به علاوه نتیجهٔ آنها در معرض دید توسعه‌دهنده (Deveploper) قرار می‌گیرد که بدین شکل پروسهٔ عیب‌یابی (Debugging) برنامه به مراتب راحت‌تر خواهد شد.

۲.۶.۱ ذخیره‌کردن خطاها در یک فایل

همان طور که در جدول زیر مشاهده می‌شود، تمامی برنامه‌هایی که در یونیکس/لینوکس اجرا می‌شوند از ۳ اصطلاحاً File Stream برخوردار خواهند بود:

جدول(۲-۷) انواع استریم‌ها در لینوکس

Stream	توضیحات	علامت
Stdin	ورودی استندرد که به‌صورت پیش‌فرض کیبورد/ترمینال است.	۰
stdout	خروجی استندرد که به‌صورت پیش‌فرض از طریق خط فرمان (CLI) نمایش داده می‌شود.	۱
Stderr	error استندرد که پیام‌های خطا نمایش داده شده یا ذخیره می‌شوند.	۲

با استفاده از مفهومی تحت عنوان redirection، می‌توانیم استریم‌ها را در یک فایل ویا فایل‌های مجزایی جهت تحلیل برنامه، ذخیره سازیم. برای روشن‌تر شدن کاربرد این استریم‌ها، فایلی تحت عنوان error.sh ساخته و کُدهای زیر را داخل آن می‌نویسیم:

```
#! /bin/bash
sum=0
for i in 1 2 3 4
do
 sum=(($sum+$ir))
done
echo “The sum of $i numbers is: $sum”
```

پس از اجرایی‌کردن این فایل، یک بار آن را تست می‌کنیم:

```
$ bash error.sh
error.sh: line 5: syntax error near unexpected token
error.sh: line 5: ` sum=(($sum+$ir))'
error.sh: line 6: syntax error near unexpected token `done'
error.sh: line 6: `done'
```

با توجه به این‌که اسکرِپت فوق حاوی مشکل است، تحت هیچ عنوان اجرا نخواهد شد و می‌بینیم که با error

مواجه شدیم. حال قصد داریم دستور دهیم تا خروجی مرتبط با errorها یا همان کُد ۲ را در فایلی تحت عنوان error.txt ذخیره سازیم:

```
$ bash error.sh 2> error.txt
```

در واقع در دستور فوق با استفاده از ۲> پس از bash error.sh دستور داده‌ایم که کلیه errorهای مرتبط با این اسکرِیپت در فایلی تحت عنوان error.txt ذخیره شوند (لازم به ذکر است با توجه به این‌که این فایل وجود خارجی ندارد، ابتدا ساخته شده سپس errorها داخل آن ذخیره می‌شوند). حال در ادامه با استفاده از دستور cat که برای نمایش دادن محتویات فایل‌ها است، محتویات ذخیره شده در فایل error.txt را نمایش می‌دهیم:

```
$ cat error.txt
error.sh: line 5: syntax error near unexpected token `('
error.sh: line 5: ` sum=(($sum+$ir))'
error.sh: line 6: syntax error near unexpected token `done'
error.sh: line 6: `done'
```

می‌بینیم همان errorهایی که در حین اجرای برنامه در ترمینال نمایش داده می‌شدند، حال در فایل مرتبط با errorها ذخیره شده‌اند.

خلاصهٔ فصل دوم

به‌طور خلاصه می‌توان گفت shell script، قابلیت برنامه‌نویسی به اضافه دستورات لینوکس است. لینوکس شل‌های مختلفی دارد ولی شل پیش‌فرض bash می‌باشد، برای مشاهدهٔ لست تمام شل‌های لینوکس می‌توانید فایل /etc/shells را مشاهده کنید.

در اسکرِپت‌نویسی Shell می‌توان خروجی را در یک فایل ذخیره کرد که به چنین کاری اصطلاحاً Output Redirection گفته می‌شود. علامت > برای ذخیره‌سازی خروجی در یک فایل مورد استفاده قرار می‌گیرد.

پروسهٔ خواندن ورودی از یک فایل اصطلاحاً Input Redirection نامیده می‌شود که برای این‌کار از علامت < استفاده می‌شود.

در شل قابلیت تعریف متغیر وجود دارد، متغیرها از نظر تعریف دو نوع هستند، یا متغیر سیستمی هستند که از قبل در سیستم‌عامل تعریف شده می‌باشد ویا متغیر کاربر است که توسط کاربر تعریف می‌شود. متغیرهای محیطی (Environment Variable) یکی از انواع متغیرهای سیستمی است که از آنها در اسکرِپت‌نویسی می‌توان استفاده کرد.

Function (تابع) مجموعه‌یی از دستورها و دستورات shell است که تحت یک نام ذخیره می‌شوند و وظیفهٔ مشخصی را انجام می‌دهند.

شل عملگرهایی برای چک‌کردن عبارت‌های شرطی، چک‌کردن فایل‌ها و فولدرها و همچنان رشته‌ها دارد، که بیشتر در ساختارهای شرطی if, elif , case استفاده می‌شود.

سه ساختار حلقه (loop) در شل قابل تعریف و استفاده است (for,while, until).

شما می‌توانید با اجرای دستور bash –x ./script_file حالت Debug Mode (وضعیت دیباگینگ) را فعال نمایید و برای عیب‌یابی کُدهایی اسکرِپت خود اقدام کنید.

سوالات و فعالیت های فصل دوم

۱. Sehll script چیست و چطور می‌توان یک اسکرِپت را نوشت و اجرا کرد؟
۲. پنج متغیر محیطی را نام ببرید و بیان کنید که چه طور می‌توان یک متغیر جدید در شل تعریف کرد.
۳. با چه دستوری می‌توان خروجی یا نتیجهٔ اجرای یک اسکرِپت را مشاهده کرد؟
۴. چند نمونه از دستورات built-in شل را نام ببرید.
۵. چهار عملگر شرطی برای کار با فایل‌ها را نام برده و کارکرد آنها را تشریح کنید.
۶. چطور یک تابع را می‌توان تعریف کرد؟
۷. به چند طریق می‌توان محاسبات ریاضی را در شل اجرا کرد؟ برای هر یک مثال بزنید.
۸. چطور می‌توان نتیجهٔ خطاهای یک اسکرِپت را در یک فایل ذخیره کرد؟

فعالیت ها

۱. یک متغیر در شل با نام دلخواه بسازید و یک قیمت را به آن نسبت داده و ذخیره کنید.
۲. متغیر را در لست متغیرهای محیطی سیستم‌عامل به ثبت برسانید (با استفاده از دستور export)
۳. ترمینال را بسته کنید و مجدد باز کنید. آیا متغیر شما در لست متغیرهای محیطی است؟
۴. یک اسکرِپت بسازید که دو عدد را از ورودی بگیرد و اگر قابل تقسیم باشند نتیجهٔ آنها را نمایش دهد و اگر قابل تقسیم نباشند (یعنی مخرج صفر باشد) پیغام خطایی را نشان دهد که مخرج نباید صفر باشد.
۵. یک اسکرِپت بسازید که از شما یک عدد را گرفته و به همان تعداد دایرکتری در یک دسکتاپ شما با نام‌های d1,d2,d3 ,... بسازد.
۶. یک اسکرِپت بنویسید که از شما آدرسی از فایل‌سیستم را بگیرد (آدرس فایل یا آدرس یک دایرکتری) و تشخیص دهد که این آدرس فایل است یا دایرکتری، اگر فایل باشد پیغام file و اگر دایرکتری باشد پیغام directory و اگر آدرس اشتباه باشد پیغام error را نشان دهد.

فصل سوم

مدیریت کاربران در سیستم‌عامل سرور

هدف کلی: با نحوهٔ تنظیم حساب کاربری وگروپ و تعریف حق دسترسی در لینوکس آشنا شوند.

اهداف آموزشی: در پایان این فصل محلان قادر خواهند شد تا:

۱. حساب کاربر (User Account) و گروپ را تشریح نمایند.
۲. حساب کاربر (Uesr account) را ایجاد و حذف کرده بتوانند.
۳. نحوهٔ ایجاد و حذف گروپ را تشریح کنند.
۴. اضافه‌نمودن حساب کاربران به یک گروپ و حذف کاربران از گروپ‌ها را انجام بدهند.
۵. راهکارهای امنیتی و تخنیک‌های ایمن‌سازی حساب کاربرها و پس‌وردها در لینوکس را پیشنهاد کنند.

در لینوکس مفهومی به نام کاربر (User) و گروپ (Group) وجود دارد. منظور از کاربر شخصی است که از کمپیوتر استفاده می‌کند و گروپ زمانی تشکیل می‌شود که خواسته باشیم سطح دسترسیِ خاصی را به گروپی از کاربران به‌طور یکسان بدهیم. در لینوکس تمام عیارسازی‌ها و وظایف به وسیلهٔ فایل انجام می‌شود. برای مدیریت بهتر منابع و تعریف سطوح مختلف دسترسی به فایل‌سیستم ضرورت به یک ساختار پالیسی منظم برای دسترسی است. به این کار مدیریت کاربران در سرور لینوکس گفته می‌شود.

۳.۱ مدیریت حساب کاربری (User account) در لینوکس

سیستم‌عامل‌های لینوکس قابلیت استفادهٔ همزمان چندین کاربر را دارد که از آن به‌عنوان multi-user یاد می‌شود. یعنی همزمان چندین کاربر مختلف می‌تواند به سیستم‌عامل وارد شده و با سیستم‌عامل کار کند و در عین حال هر کدام از آنها قادر خواهند بود تا یک محیط کاربری شخصی‌سازی‌شده[۱] را داشته باشند. در ادامه با نحوهٔ حذف و اضافه‌نمودن حساب کاربرها و گروپ‌ها، استفاده از کلیدهای میانبر[۲]، تنظیم‌کردن مجوزهای دسترسی روی فایل‌ها و غیره آشنا خواهیم شد.

با استفاده از دستورات whoami و who می‌توانیم به معلومات کاربر/کاربرانی که در سیستم وارد شدند دسترسی پیدا کنیم؛ به‌طور مثال: با واردکردن دستور who در ترمینال، با خروجی زیر مواجه خواهیم شد:

```
Afghanacademy tty7 2016-08-24 10:20
```

این دستور، کلیهٔ کاربرانی را که در این لحظه در سیستم وارد گردیده اند، به ما نشان خواهد داد که در این مثال فقط یک کاربر با حساب کاربری تحت عنوان Afghanacademy را مشاهده می‌کنیم. درصورتی‌که بخواهیم معلومات تکمیلی از کاربران داشته باشیم، می‌توانیم از آپشن a- پس از دستور who استفاده نماییم:

```
who -a
```

خروجی دستور فوق:

```
system boot 2016-08-24 14:49
Afghanacademy + tty7 2016-08-24 10:20 old 3224 LOGIN tty1 2016-08-24 10:20 3299
id=tty1 run-level 5 2016-08-24 10:23
```

دستور whoami هم که برگرفته از عبارت انگلیسی ?Who am I، معلوماتی دربارهٔ کاربر فعلی را که وارد سیستم شده است، نشان می‌دهد:

```
whoami
```

خروجی دستور فوق:

---

[۱] - customized

[۲] - short-cut

```
Afghanacademy
```

با استفاده از دستور id نیز می‌توان به معلومات تکمیلی در مورد کاربری که در حال حاضر در سیستم وارد شده است، دست یابیم:

```
id
```

خروجی دستور فوق:

```
uid=1000(Afghanacademy)                    gid=1000(Afghanacademy)
groups=1000(Afghanacademy),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),11
2(lpadmin),124(sambashare),129(debian-tor)
```

خروجی دستور فوق نشان می‌دهد که کاربر فعلی وارد شده دارای شناسهٔ کاربری (user id) ۱۰۰۰ است که در ادمه بیشتر با uid، gid و... آشنا خواهیم شد.

۳.۱.۱ حذف و اضافه‌نمودن کاربر

توزیع‌های مختلف از طریق GUI امکاناتی را برای حذف و اضافه‌نمودن کاربران و گروپ‌ها دارند؛ اما در عین حال امکان حذف و اضافه‌نمودن حساب کاربرها از طریق خط فرمان (CLI) نیز وجود دارد؛ البته با اشاره به این نکته که اجرای این دستورات از صلاحیت‌های کاربر root می‌باشد و کاربران عادی این صلاحیت را ندارند.

با استفاده از دستور useradd می‌توان یک کاربر جدید ایجاد کرد و درصورتی‌که بخواهیم یک کاربری را که قبلاً ایجاد شده، حذف نماییم، می‌توانیم از دستور userdel استفاده نماییم. فرض کنیم قصد داریم یک حساب کاربر جدید تحت عنوان user۱Admin ایجاد کنیم؛ ساده‌ترین دستور برای انجام دستور زیر می‌باشد:

```
sudo useradd user1Admin
```

با اجرای این دستور علاوه بر ساخت یک حساب کاربر برای کاربر جدید، یک دایرکتری به نام خود کاربر در مسیر /home/user1Admin ساخته می‌شود. که مثل my document در سیستم‌عامل ویندوز برای هر کاربر می‌ماند. این دایرکتری برای ذخیره‌کردن فایل‌های شخصی کاربر، از جمله فایل‌های Desktop, Documents , Download , ... استفاده می‌شود. بعد از ایجاد کاربر، یک حساب کاربری برای آن ساخته می‌شود و نوبت به دادن پس‌ورد می‌رسد، برای دادن پس‌ورد از دستور passwd به شکل زیر استفاده می‌شود:

برای دادن پس‌ورد از دستور passwd به شکل زیر استفاده می‌شود:

```
sudo passwd user1Admin
```

بعد از اجرای دستور فوق، پیغام: Enter new UNIX password نشان داده می‌شود؛ رمزعبور مد نظر

خود را وارد نموده و انتر نمایید. پیغام: Retype new UNIX password نمایش داده می‌شود، رمزعبوری را که قبلاً وارد کرده بودید، دقیقاً به همان شکل باید وارد کنید. درصورتی‌که هر دو رمزعبور یکسان باشند، پیغام passwd: password updated successfully نمایش داده می‌شود. حال با استفاده از دستور id که پیش از این با آن آشنا شدیم، قصد داریم معلومات این کاربر جدید را مشاهده کنیم:

```
id user1Admin
```

خروجی دستور فوق

```
uid=1001(user1Admin) gid=1001(user1Admin) groups=1001(user1Admin)
```

می بینیم که شناسهٔ (ID)۱۰۰۱ برای این کاربر جدید در نظر گرفته شده است. برای حذف این حساب کاربر هم به سادگی می‌توانیم دستور زیر را در دستور لاین اجرا نماییم:

```
sudo userdel user1Admin
```

دستور فوق دایرکتری home این کاربر را حذف نمی‌کند و درصورتی‌که بخواهیم دایرکتری home کاربر را نیز حذف نماییم، باید از آپشن r- پس از دستور userdel استفاده کنیم. درصورتی‌که مجدداً دستور id user1Admin را در خط فرمان (CLI) اجرا کنیم، از آنجاکه چنین کاربری دیگر وجود ندارد، خروجی زیر نشان داده می‌شود:

```
id: ‘user1Admin’: no such user
```

۳.۱.۲ حساب کاربر کاربری root

همان‌طور که پیش این هم توضیح داده شد، حساب کاربر root دارای بیشترین صلاحیت‌ها در سیستم است لذا حساب کاربری بسیار «قدرتمند» محسوب می‌گردد. سایر سیستم‌عامل‌ها همچون ویندوز حساب کاربری را که دارای تمامی صلاحیت‌ها است، administrator می‌نامند اما در لینوکس چنین حساب کاربری superuser یا root user نامیده می‌شود.

هشدار: پیش از دادن صلاحیت‌های روت به یک کاربر، از مطمئن‌بودن وی اطمینان حاصل کنید چرا که یک کاربر با سطح دسترسی root امکان انجام هر کاری را خواهد داشت (حتی محدودکردن دسترسی خود شما که این حساب کاربر را برایش ساخته‌اید).

۳.۲ حذف و اضافه‌نمودن گروپ

لینوکس از گروپ برای مدیریت کاربران استفاده می‌کند. گروپ مجموعه‌یی از حساب کاربرهایی هستند که صلاحیت‌های یکسان دارند. مدیریت گروپ‌ها در لینوکس از طریق فایلی که در مسیر /etc/group قرار دارد صورت می‌گیرد؛ به عبارت دیگر، گروپ‌ها در لینوکس به‌منظور ایجاد دسته‌بندی کاربران که دارای نیازهای یکسان، اهداف یکسان، مجوزهای یکسان و مسایل امنیتی مشابه می‌باشند، مورد استفاده قرار

می‌گیرند و به‌صورت پیش‌فرض، هر کاربری که در لینوکس ساخته می‌شود به یک گروپ پیش‌فرض یا اصلی تعلق دارد (Ward,۲۰۱۴).

تمامی کاربران سیستم‌عامل لینوکس یک شناسه (id) تحت عنوان uid دارند که یک عدد صحیح - مثلاً ۱۰۰۰- است؛ علاوه بر این، هر کاربر یک شناسهٔ گروپ هم تحت عنوان gid نیز دارند. با استفاده از دستور groupadd می‌توان اقدام به ایجاد یک گروپ جدید در لینوکس کرد:

```
groupadd admins
```

حال اگر نگاهی به داخل فایل /etc/group بیندازیم، خواهیم دید ک در خط آخر گروپی تحت عنوان admins اضافه شده است.

نکته: همان طور که پیش از این بیان شد، برای حذف/اضافه‌نمودن کاربر یا گروپ، نیازمند حق دسترسی root می‌باشیم، برای اجرای این دستورات دو راه وجود دارد، یا باید از طریق حساب کاربر root به سیستم وارد شویم ویا این‌که موقتی به حساب کاربر root سویچ نماییم که این کار را از طریق دستورهای su و sudo قبل از هر دستور یا دستور می‌توان انجام داد.

درصورتی‌که بخواهیم گروپ مد نظر را حذف نماییم، می‌توانی‌م از دستور groupdel استفاده نماییم:

```
groupdel admins
```

درصورتی‌که مجدداً فایل /etc/group را مشاهده کنیم، خواهیم دید که دیگر گروپی تحت عنوان admins وجود ندارد. حال سؤالی که ممکن است پیش بیاید این است که چگونه یک کاربر را می‌توان به گروپی که ایجاد کرده‌ایم، اضافه نماییم؟ در پاسخ به این سؤال باید گفت که دستور usermod این کار را به سادگی برایمان انجام خواهد داد. برای روشن‌تر شدن این مسأله، ابتدا یک گروپ جدید تحت عنوان developers ایجاد می‌کنیم:

```
groupadd developers
```

سپس با استفاده از دستور زیر کاربری جدید تحت عنوان user1Admin ایجاد می‌کنیم:

```
useradd user1Admin
```

حال با استفاده از دستور زیر می‌توانیم کاربر user1Admin را به گروپ developers اختصاص دهیم:

```
usermod -G developers user1Admin
```

در لینوکس دستوری تحت عنوان groups وجود دارد که توسط آن می‌توان گروپ‌هایی را که یک کاربر به

آنها تعلق دارند مشاهده کرد، برای روشن‌تر شدن این موضوع به مثال زیر توجه کنید، برای آن که مطمئن شویم کاربر user1Admin به گروپ developers تعلق دارد یا خیر، دستور زیر را اجرا کنید:

```
groups user1Admin
```

خروجی دستور فوق:

```
user1Admin: user1Admin developers
```

درصورتی‌که بخواهیم کاربر user1Admin را از گروپ developers حذف کنیم، دستور زیرا را باید وارد ترمینال کنیم:

```
usermode -G user1Admin user1Admin
```

حال اگر دستور groups user1Admin را در ترمینالی اجرا کنیم که برای نشان‌دادن گروپ‌هایی است که یک کاربر به آنها تعلق دارد، با خروجی زیر مواجه خواهیم شد:

```
user1Admin: user1Admin
```

می بینیم که این کاربر از گروپ کاربران developers حذف شده و فقط به گروپ کاربری user1Admin که به‌صورت پیش‌فرض در حین ساخت این حساب کاربر جدید ایجاد شده تعلق دارد.

۳.۳ معلومات حساب کاربران

در لینوکس، هویت هر کاربر با یک شناسهٔ کاربری تحت عنوان uid (گرفته شده از User ID به معنای شناسهٔ کاربری) مشخص می‌گردد؛ و دیتابیسی در سیستم‌عامل لینوکس وجود دارد که Username هر کاربر به همراه uid وی در آن به ثبت می‌رسد. زمانی‌که در لینوکس یک کاربر جدید ایجاد می‌شود، معلومات وی در این دیتابیس اضافه شده و یک دایرکتری home برای این کاربر جدید ایجاد می‌گردد.

معلومات کاربران در فایلی تحت عنوان passwd که در مسیر /etc/passwd قرار دارد، ذخیره می‌شود، این فایل شامل معلوماتی می‌باشد که در جدول زیر آمده است (Ward,۲۰۱۴):

جدول(۳-۱) معلومات کاربران در فایل/etc/passwd

بخش	جزئیات	تشریحات
Username	نام کاربر برای واردکردن به سیستم	باید بین ۱ الی ۳۲ کرکتر طول داشته باشد
Password	رمزعبور کاربر که با حرف x نمایش داده می‌شود و رمزگذاری شده است.	رمزعبور زمانی‌که در لینوکس تایپ می‌شود به دلایل امنیتی هیچ وقت نمایش داده نمی‌شود.

User ID (UID)	شناسهٔ کاربری آی دی مخصوص هر کاربر که تکراری نیست	شناسهٔ ۰ برای کاربر روت در نظر گرفته شده است. کاربران معمولی از شناسه‌های ۱۰۰۰ به بالا دارند به جز توزیع RHEL که از عدد ۵۰۰ شروع می‌شود.
Group ID (GID)	شناسهٔ گروپ	این شناسه در مسیر /etc/group ذخیره می‌گردد.
User Info	این فیلد اختیاری است و این امکان را به ما می‌دهد تا معلومات بیشتری در مورد یک کاربر ذخیره نماییم.	
Home Directory	مسیر قرارگیری دایرکتری home کاربر	برای مثال /home/Afghanacademy
Shell	مسیر قرارگیری Shell پیش‌فرض کاربر	

اگر بخواهیم فیلدهای فوق‌الذکر را در کنار هم داشته باشیم، می‌توانیم نمونهٔ یوزر روت زیر را در نظر بگیریم:

root:x:0:0:root:/root:/bin/bash

۳.۳.۱ انواع حساب‌های کاربری

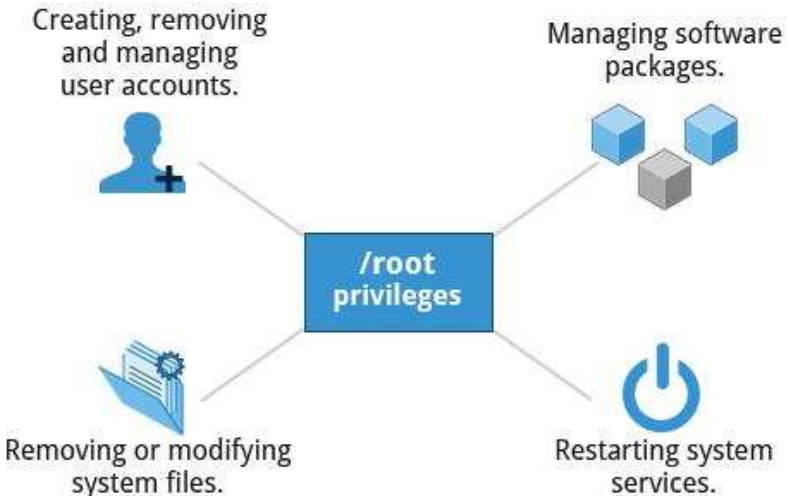
به‌صورت پیش‌فرض، لینوکس دارای ۴ نوع حساب کاربری به شرح زیر است:

- روت؛
- سیستم؛
- نرمال؛
- شبکه.

از نظر امنیت توصیه می‌شود که تا حد امکان حق دسترسی حساب کاربران را در حد صلاحیت آنها محدود بسازید و حساب کاربرهای غیر فعال را حذف نمایید.

نکته: برای آگاهی از آخرین تایمی که کاربر به سیستم وارد شده است، می‌توانیم از دستور last استفاده نماییم.

به‌طور کلی، حساب کاربر root دارای بیشترین صلاحیت‌ها است؛ با استفاده از حساب کاربر روت می‌توان به مدیریت کامل سیستم پرداخت از آن جمله می‌توان به حذف/اضافه‌نمودن حساب کاربرهای جدید، تغییر پس‌ورد، نصب نرم‌افزار، تغییر فایل‌های سیستمی، راه‌اندازی سرویس‌های سیستمی و غیره اشاره کرد.



شکل (۳-۱) صلاحیت‌های حساب کاربر root

زمانی‌که ما با حساب کاربر root در سیستم وارد می‌شویم، در آخر خط فرمان (CLI) علامهٔ # نمایش داده می‌شود که نشانهٔ حساب کاربر root است:

root@Afghanacademy-inspiron-1545:/#

۳.۴ مقایسهٔ دستورات sudo و su

در لینوکس برای کاربران عادی که روت نیستند این امکان فراهم شده تا در صورت نیاز، بتوانند به‌صورت موقت از صلاحیت‌های root استفاده کنند و کارهایی را که نیاز به سطح دسترسی root می‌باشد عملی سازند. برای این کار، ۲ دستور مختلف وجود دارد که عبارتند از: sudo و su که تفاوت‌های این ۲ دستور در جدول زیر ارائه شده است(Ward,۲۰۱۴) :

جدول (۳-۲) مقایسه دستور su و sudo

Sudo	su
زمانی‌که با استفاده از این دستور قصد داریم صلاحیت‌های root را به کاربری بدهیم، ضرورت به واردکردن پس‌ورد کاربر داریم نه پس‌ورد root	زمانی‌که با استفاده از این دستور قصد داریم صلاحیت‌های root را به کاربری بدهیم، ضرورت به واردکردن پس‌ورد root است. دادن پس‌ورد root به کاربران عادی توصیه نمی‌شود
این دستور قابلیت‌های بیشتری دارد و ایمن‌تر است.	زمانی‌که کاربری با استفاده از دستور su به صلاحیت‌های root دسترسی پیدا کند، این کاربر امکان انجام هر کاری را خواهد داشت.
این دستور قابلیت‌های زیادی داراست.	این دستور قابلیت‌های کمتر دارد

دستور sudo این امکان را می‌دهد تا بتوان رفتار کاربرانی را که از این دستور استفاده نموده اند، زیر نظر بگیریم؛ به عبارت دیگر، اگر کاربری با استفاده از دستور sudo اقدام به گرفتن صلاحیت‌های root کند اما پس‌ورد وارد شده توسط وی اشتباه باشد، پیغامی مشابه پیغام زیر در فایل لاگ (File Log)های سیستمی ذخیره می‌شود:

```
authentication failure; logname=op uid=0 euid=0 tty=/dev/pts/6 ruser=op rhost=
user=op
conversation failed
auth could not identify password for [op]
op: 1 incorrect password attempt ;
TTY=pts/6 ; PWD=/var/log ; USER=root ; COMMAND=/bin/bash
```

همان‌طور که می‌بینیم، تمام جزئیات در اختیار ما قرار خواهد گرفت.

۳.۵ مسایل امنیتی

سیستم‌عامل لینوکس نسبت به سایر سیستم‌عامل‌ها ایمن‌تر است چرا که در این سیستم‌عامل هر پراسس (Process) از سایر پراسس‌ها مجزا است. به عبارت دیگر، در لینوکس یک پراسس نمی‌تواند به منابع سایر پراسس‌ها دسترسی پیدا کند. به همین دلیل ویروس‌ها به سختی خواهند توانست در این سیستم‌عامل به منابع سیستم دسترسی پیدا کنند.

۳.۵.۱ دسترسی به سخت‌افزار

در لینوکس همان‌طور که کاربران برای دسترسی به فایل‌ها و دایرکتوری‌های مختلف ضرورت به حق دسترسی دارند، برای استفاده از سخت‌افزار نیز حق دسترسی ضرورت دارند.

هر سخت‌افزار در لینوکس دارای یک فایل اختصاصی است که تحت عنوان Device Node شناخته می‌شود که در مسیر /dev قرار دارند. هر فایل اختصاصی از سخت‌افزارها دارای صلاحیت‌های کاربر، گروپ کاربری و سایر صلاحیت ها است؛ برای مثال: هارددسک‌ها تحت عنوان /dev/sd* شناخته می‌شوند.

۳.۵.۲ به‌روزرسانی سیستم

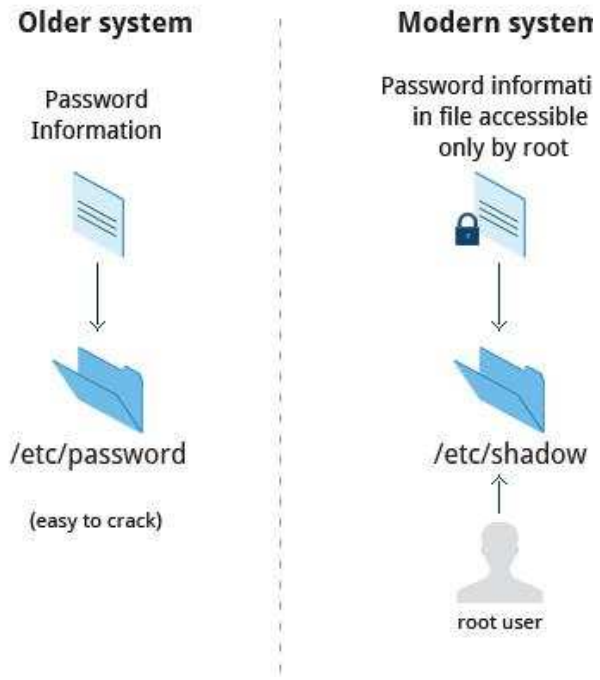
زمانی‌که یک مشکل امنیتی در کرنل لینوکس ویا برنامه‌های مختلف لینوکس یافت می‌شود، خیلی سریع این مشکلات امنیتی توسط توسعه‌دهندگان تشخیص داده شده و از طریق اعلان‌ها به اطلاع کاربران لینوکس می‌رسد که می‌باید سیستم خود را به‌روزرسانی کنند، لذا توصیه می‌شود تا از طریق به‌روزرسانی خودکار سیستم‌عامل، در دوره‌های زمانی کوتاه‌مدت سیستم خود را به‌روز نگاه داشته تا هدف حمله هکرها قرار نگیرد.

۳.۵.۳ نحوهٔ ی ذخیره‌سازی پس‌وردها در لینوکس

در گذشته، پس‌وردهای کاربران در مسیر /etc/passwd ذخیره می‌شد که این فایل توسط هر کسی قابل دسترسی بود و همین مسأله باعث شده بود تا پس‌وردها به سادگی هک (Hack) شوند. اما در سیستم‌عامل‌های جدید لینوکس، پس‌وردها در قالب یک فرمت رمزنگاری‌شده در فایل تحت عنوان /etc/shadow ذخیره می‌شوند که فقط با مجوز root می‌توان به این فایل دسترسی پیدا کرد.

۳.۵.۴ رمزنگاری پس‌ورد

در زمینهٔ امنیت سیستم‌عامل، رمزنگاری پس‌وردها گامی مهم و مؤثر است؛ بسیاری از توزیع‌های لینوکس از الگوریتم رمزنگاری تحت عنوان SHA-512 که توسط آژانس امنیت ملی ایالات متحدهٔ آمریکا (NSA) برای رمزنگاری پس‌وردهای حساس طراحی شده بهره می‌گیرند و این درحالی‌است که الگوریتم SHA-512 در پروتوکول‌ها و اپلیکیشن (Application)هایی که نیاز به رمزنگاری دیتاها دارند، به دفعات مورد استفاده قرار می‌گیرد (Ward, ۲۰۱۴).



شکل (۳۰-۲) امنیت پس‌وردها در سیستم‌های قدیم و جدید لینوکس

به‌طور مثال، اگر بخواهیم کلمهٔ test را با استفاده از این الگوریتم رمزنگاری کنیم، با خروجی زیر رو به رو می‌شویم:

ee26b0dd4af7e749aa1a8ee3c10ae9923f618980772e473f8819a5d4940e0db27ac185f8a0e1d5f84f88bc887fd67b143732c304cc5fa9ad8e6f57f50028a8ff

۳.۵.۵ راهکارهای ایجاد پس‌وردهای ایمن

راهکارهای متعددی برای ایمن ساختن سیستم‌عامل و حفاظت از معلومات وجود دارد که در ادامه به چند مورد آن اشاره می‌شود:

عمر پس‌ورد: روشی است که از آن طریق می‌توان دوره‌های زمانی مشخصی را تنظیم کرد تا از آن طریق به کاربر یادآوری شود که موقع تغییر پس‌ورد است. با اتخاذ چنین رویکردی، حتی اگر پس‌وردهای یک سیستم آشکار شود، فقط برای مدت زمان کوتاهی در دسترس خواهد بود. برای این کار از ابزاری به نام chage می‌توان استفاده کرد.

پس‌وردهای قوی: راهکار دیگری که می‌توان برای افزایش امنیت معلومات به کار برد، استفاده از پس‌وردهای قوی است. ابزاری همچون PAM این امکان را در اختیار کاربران لینوکس قرار می‌دهد تا در حین انتخاب پس‌ورد با استفاده از دستور passwd، پس‌وردهای ایمن انتخاب نمایند.

خلاصهٔ فصل سوم

- برای مدیریت بهتر منابع و تعریف سطوح مختلف دسترسی به فایل‌سیستم ضرورت به یک ساختار پالسی منظم برای دسترسی است. به این کار مدیریت کاربران در سرور لینوکس گفته می‌شود.
- بسیاری از توزیع‌های لینوکس از الگوریتم رمزنگاری تحت عنوان SHA-۵۱۲ برای ذخیرهٔ معلومات حساس استفاده می‌کنند.
- معلومات حساب کاربرها در لینوکس در دو فایل /etc/passwd و /etc/shadow ذخیره می‌شوند.

سوالات و فعالیت های فصل سوم

۱. دستورات su , sudo چه تفاوت‌هایی دارند.
۲. چه معلومات در فایل /etc/passwd برای یک کاربر ذخیره می‌شود؟

فعالیت ها

۱. با استفاده از کاربر root به سیستم‌عامل وارد کنید.
۲. ترمینال را باز کنید.
۳. در دسکتاپ خود یک دایرکتری جدید بسازید.
۴. مجوز دایرکتری جدید ساخته شده چیست؟
۵. با یک کاربر غیر از کاربر root وارد سیستم شوید.
۶. آیا می‌توانید به دایرکتری /root وارد شوید؟ چرا؟ چه راهی برای واردشدن به این دایرکتری وجود دارد؟
۷. با معلوماتی که از فصل قبل در مورد اسکرِپت آموخته‌اید، اسکرِپتی بنویسید که با اجرای آن یک عدد؛ مثلاً: ۱۰ را از ورودی بگیرد و به همان تعداد کاربر جدید با نام‌های user۱,user۲,..., user۱۰ بسازد.
۸. اسکرِپتی بنویسید که با اجرای آن تمام حساب کاربرهای ساخته‌شدهٔ قبلی را پاک کند.

فصل چهارم

راه‌اندازی سرویس DNS

هدف کلی: محصلان با روش کار و و نحوهٔ تنظیم DNS سرورها در لینوکس آشنا شوند.

اهداف آموزشی: در پایان این فصل محصلان قادر خواهند شد تا:

۱. مفاهیم اولیه و اصطلاحات مورد استفاده در DNS سرورها را تشریح نمایند.
۲. نحوهٔ کار و ساختار طراحی DNS سرورها را توضیح دهند.
۳. مهارت نصب و فعال‌سازی سرویس DNS در لینوکس را کسب نمایند.
۴. مهارت عیارسازی سرویس Bind به‌عنوان یک سرویس DNS در لینوکس را کسب نمایند.
۵. انواع سرورهای DNS و نحوهٔ عیارسازی آنها تشریح کنند.

DNS[1] یا سیستم نام دامنه یکی از بخش‌های مشکل عیارسازی وب‌سایت‌ها و سرورها است. فهم نحوهٔ کار DNS به شما کمک می‌کند که مشکلات مربوط به دسترسی به وب‌سایت‌ها را رفع کنید و همچنین درک شما از آنچه در پشت پرده رخ می‌دهد را افزایش می‌دهد. پیش از آن که وارد مبحث عیارسازی DNS سرورها شویم، به معرفی نحوهٔ کار و اصطلاحات تخصصی آن می‌پردازیم.

۴.۱ درک ساختار DNS

ساختار DNS و خود DNS جهت سازماندهی و پیداکردن آدرس IP کمپیوترها در یک شبکه توزیع‌شدهٔ کلان مثل انترنت، معرفی و توسعه داده شده است. قبل از هر چیز به فلسفهٔ اصلی به‌وجودآمدن DNS اشاره می‌کنیم، دلیل اصلی به‌وجودآمدن DNS این است که کارکردن و به خاطرسپردن نام‌ها برای انسان آسانتر از کارکردن و به خاطرسپردن اعداد مثل IP است، امروزه شما برای بازکردن سایت‌های مورد علاقهٔ خود هیچ وقت IP آنها را به خاطر نمی سپارید بلکه نام‌های آن را به خاطر دارید مثل googl.com که از طریق IP های مختلف قابل دسترس است، شما همیشه نام یا آدرس انترنتی آن را وارد می‌کنید، ولی در پشت صحنه این نام توسط DNS سرورها به یک IP تبدیل می‌شود، پس اگر DNS سرورها نبود شما دیگر با نام‌های انترنتی دیگر نمی توانستید به سرویس‌های انترنت دسترسی پیدا کنید(Sobell,۲۰۱۴) .

در نبود DNS سرورها، نام کمپیوترها و IP آدرس متناظر آنها در یک فایل متنی ساده به نام hosts ذخیره می‌شد، که هنوز هم در سیستم‌عامل‌های مختلف این فایل وجود دارد. این فایل توسط هر کاربر به‌طور جداگانه در کمپیوترش نگهداری می‌شود. در سیستم‌عامل لینوکس این فایل در آدرس /etc/hosts موقعیت دارد. پس با استفاده از این فایل، بدون این‌که یک DNS سرور داشته باشید نیز می‌توانید یک نام را به یک IP نسبت دهید. این به این معناست که سیستم‌عامل ابتدا نام را در این فایل جستجو می‌کنند و اگر نام در این فایل یافت شود آن را به IP متناظر آن تبدیل می‌کند و به DNS سرور برای ترجمه روان نمی‌کند، نام‌های انترنتی را «دومین» (Domain) هم می‌گویند. شکل زیر نمونه‌یی از این فایل را نشان می‌دهد (Soyinka,۲۰۱۶).

```
# Host table for Internal network
127.0.0.1     localhost.localdomain  localhost
::1           localhost6.localdomain6  localhost6
192.168.1.1   serverA.example.org   serverA # Linux server
192.168.1.2   serverB.example.org   serverB # Other Linux server
192.168.1.7   dikkog                       # Win2003 server
192.168.1.8   trillion                     # Cluster master node
192.168.1.9   sassy                        # FreeBSD box
10.0.88.20    laserjet5                    # Lunchroom Printer
```

شکل (۴-۱) محتوای فایل /etc/hosts در سیستم‌عامل لینوکس

[1] - Domain Name System

این فایل یک ساختار ساده دارد و سه ستون از معلومات را در خود نگهداری می‌کند. ستون اول IP است، ستون دوم «دومین» است و ستون سوم نام کوتاه یا نام خلاصه است که اختیاری می‌باشد. بین این ستون‌ها فاصله یا tab وجود دارد. برای اضافه‌کردن تشریحات از علامت # قبل از تشریحات استفاده می‌شود.

این فایل برای شبکهٔ محلی با تعدادی محدودی از کمپیوترها با نام‌های مشخص مناسب است اما برای تعداد زیادی از سیستم‌ها و کمپیوترها در شبکه‌های مختلف دیگر کارآمد نیست. برای روشنی بیشتر یک مثال می‌آوریم. فرض کنید دو mail سرور در دو شبکهٔ مجزا داشته باشیم. فایل hosts در شبکهٔ اول شامل معلومات زیر است:

192.168.0.2 mailserver

و فایل hosts در شبکهٔ دوم نیز شامل معلومات زیر باشد:

192.168.1.2 mailserver

مدیران شبکه در هر دو شبکه نام‌های یکسان برای mail سرورها در نظر گرفته اند که هر کدام از آنها کمپیوترهای متفاوتی هستند. اگر هر دو شبکه را با هم ادغام کنیم و یک فایل hosts واحد برای ترکیب هر دو شبکه بخواهیم در نظر بگیریم، با مشکل نام‌های تکراری مواجه می‌شویم، یک راه حل تغییر نام یکی یا هر دوی mail سرورها می‌باشد، در این صورت فایل hosts ما به‌صورت زیر خواهد بود.

192.168.0.2 mailserver1

192.168.1.2 mailserver 2

راه دیگر آن است که یک ساختار برای نامگذاری استفاده کنیم که دیگر نیاز به تغییر نام و انتخاب نام‌های متفاوت در شبکه‌های مختلف نباشد، برای انجام این کار بهتر است نام شبکه را نیز به‌عنوانی بخشی از ساختار نامگذاری اضافه کنیم. در این صورت برای هر دو mail سرور خواهیم داشت:

mailserver in network1

mailserver in network2

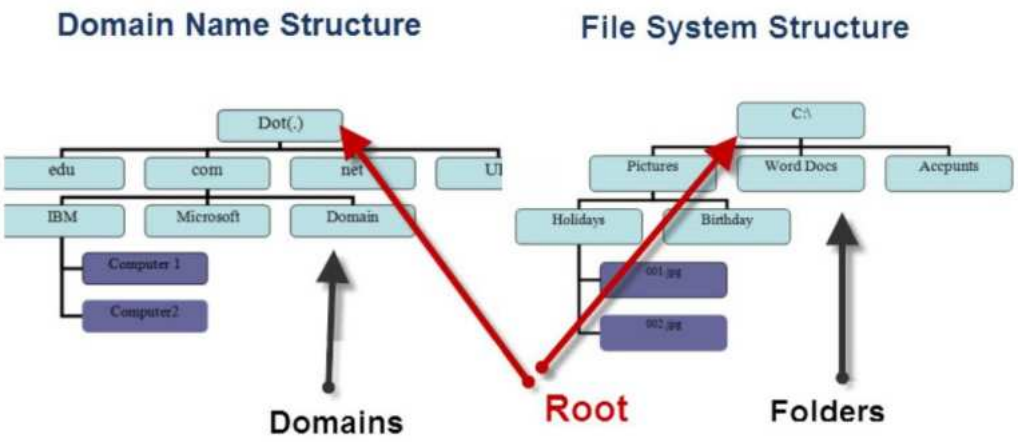
به جای نام شبکه از اصطلاح دومین (domain) در انترنت استفاده می‌شود پس خواهیم داشت:

mailserver in domain1

mailserver in domain2

این ساختار نامگذاری بهتر از حالت اولیه است و برای شبکه‌های کوچک با تعداد کمی کمپیوتر جواب می‌دهد ولی وقتی که هزاران سرور و شبکه‌های متصل به هم داشته باشید (مثل انترنت) دیگر جوابگو نیست. مدیریت معلومات تمام دومین (Domain)های شبکه در یک فایل و به‌روزرسانی‌کردن و انتشار تغییرات آن در تمام شبکه چالش‌هایی است که دیگر با فایل hosts نمی‌توان آن را برطرف کرد.

برای شبکه‌های بزرگ یک ساختار سلسله‌مراتبی یا درختی به وجود آمد. این ساختار تا حد زیادی مشابه ساختار درختی سیستم‌فایل در سیستم‌عامل لینوکس و ویندوز است. همان‌طور که در سیستم‌فایل فایل‌ها داخل دایرکتری‌ها قرار دارند و یک دایرکتری می‌تواند چندین دایرکتری دیگر را داشته باشد، در ساختار DNS سرورها نیز تمامی IP و نام کمپیوترها (مانند فایل‌ها) در داخل یک دومین (مانند دایرکتری‌ها) قرار می‌گیرند و یک دومین می‌تواند شامل دومین‌های دیگر (subdomain) باشد. ریشه یا رأس این ساختار نقطه (.) است. به عکس زیر توجه کنید:



شکل (۴-۲) شباهت سیستم‌فایل و ساختار DNS

در سیستم‌فایل نامگذاری آدرس‌ها از بالا به پایین انجام می‌شود مثل /home/admin/documents ولی در DNS نامگذاری از پایین به طرف ریشه است مثل computername.subdomain.domain.، علامت جدا کننده \ یا / است ولی در DNS علامت جداکننده نقطه (.) است.

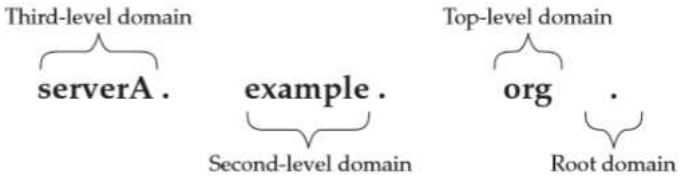
۴.۲ سطوح مختلف دومین

ما معمولاً سایت‌ها را از طریق آدرس انترنتی آنها که به آن آدرس FQDN[۱] نیز گفته می‌شود دسترسی و پیدا می‌کنیم؛ مثل: serverA.example.org هر یک از اجزای این آدرس معنای خاصی می‌دهد. همان‌طور که قبلاً بیان شد، ساختار DNS یک ساختار درختی است.

آدرس‌های انترنتی عموماً از سه سطح دومین تشکیل شده‌اند:

- دومین ریشه (Root domain)
- دومین سطح بالا (Top-Level Domain)
- دومین سطح دوم (Second-Level Domain)

[۱]- - fully qualified domain name



شکل(۴-۳) سطوح مختلف دومین‌ها

۴.۲.۱ دومین ریشه (Root Domain)

این دومین‌ها در بالاترین سطح قرار دارند. در اصل زمانی‌که شما یک وب‌سایت را در مرورگر باز می‌کنید، هیچ وقت نقطهٔ آخری را در آدرس یک وبسایت وارد نمی سازید. ولی به‌طور مخفیانه آن را در نظر می‌گیرد ولی برای شما نمایان نیست. نقطهٔ آخری در اصل نشانهٔ root domain است. دومین ریشه در اصل اشاره به ۱۳ سرور نام ریشه (root name server) دارند که در نقاط مختلف جهان پراگنده شده‌اند. شما می‌توانید آنها را به‌عنوان مغز انترنت در نظر بگیرید، که اگر آنها نباشند شما هیچ آدرس سایتی را باز نمی توانید. به دلیل حساسیت و اهمیت این سرورها (Backup) مختلفی از این سرورها در نقاط مختلف جهان در نظر گرفته شده تا در صورت بروز حوادثی مثل جنگ، زمین‌لزره و... انترنت از کار نیفتد. نام این سرورها با حروف الفبا شروع می‌شود. مثل a.root-server.net, b.root-server.net و... می‌باشد. سرورهای ریشه توسط سازمان‌های مختلف کنترول می‌شوند و این قابلیت کنترول از سوی ICANN (سازمان انترنتی واگذاری نام‌ها و اعداد) به این سازمانها واگذار شده است (Negus, ۲۰۱۰; Petrov, ۲۰۱۸).

۴.۲.۲ دومین سطح بالا (Top Level Domain)

این دومین‌ها بعد از دومین ریشه و یک سطح پایین‌تر از آن قرار دارند. نام‌هایی مثل com,net, org, af,us,uk , ... نام‌های سطح بالا هستند. امروزه بیش از ۸۰۰ دومین سطح بالا در انترنت وجود دارد. TLD (Top Level Domain) به دسته‌های مختلف تقسیم‌بندی می‌شوند:

- TLD عمومی (.edu,.gov,.net ,.com , org ...)
- TLD وابسته به کشورها (.af,.uk,.us, ...)
- TLDهای مخصوص برندهای خاص (.microsoft,.linux, ...)
- TLD مربوط به زیرساخت (.arpa)

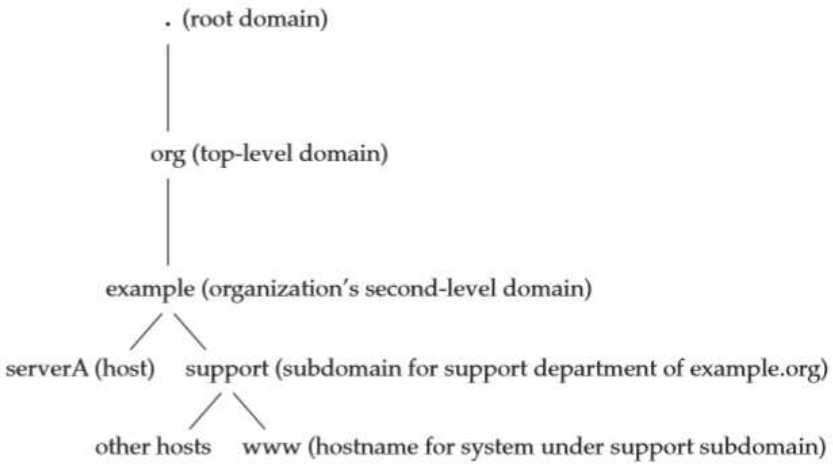
۴.۲.۳ دومین سطح دوم (second level domian)

این دومین‌ها معمولاً مربوط به یک سازمان یا شرکت خاص است و توسط ISPها مدیریت و اجاره داده می‌شود؛ برای مثال: google یک دومین سطح دوم است که مربوط به کمپنی google بوده و ثبت و راجستر شده است.

۴.۲.۴ دومین سطح سوم (Third–Level Domain)

این دومین‌ها دو نوع هستند، یا Subdomain می‌باشند ویا این‌که نام یک کمپیوتر یا سرویس واقعی در یک دومین اصلی می‌باشد. مثلاً www معمولاً نام وب‌سرور می‌باشد ویا mail در mail.google.com نیز که نام سرورهای ایمل شرکت google می‌باشند، همگی اشاره به یک سرویس یا کمپیوتر سروری ارائه‌دهندهٔ آن سرویس دارند.

اما «زیردومین» (subdomain) که جزئی از یک دومین اصلی می‌باشد چیست؟ زیردومین جزئی از یک دومین اصلی است که توسط همان دومین اصلی کنترول می‌شود. به هر دومینی که جزئی از یک دومین بزرگتر باشد و تحت آن کنترول و مدیریت شود، زیردومین گفته می‌شود. دومین‌های ریشه، سطح بالا و سطح دوم، هر کدام دومین‌های اصلی و جداگانه هستند که به‌طور مستقل تعریف می‌شوند و هیچ کدام بر دیگری کنترول ندارند ولی یک دومین بالای زیردومین خود کنترول دارد؛ برای نمونه: ما می‌توانیم یک زیردومین برای دیپارتمنت تاریخ در وب‌سایت مکتب خود اضافه کنیم www.history.school.edu در این صورت history یک زیردومین گفته می‌شود. شکلی که در ادامه آمده است بهتر مفهوم زیردومین را نشان می‌دهد، در شکل زیر support یک زیردومین است ولی server یک host است.



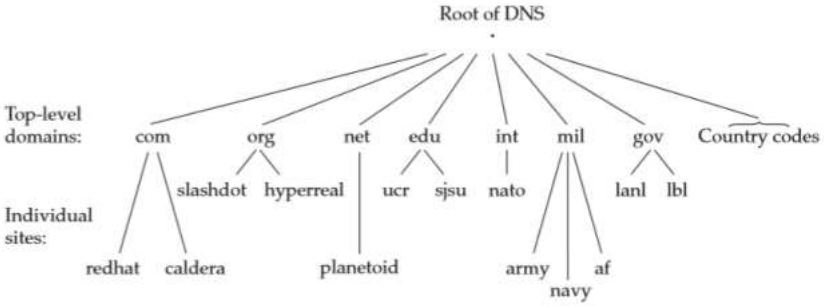
شکل (۴-۴) مفهوم زیردومین

فرق زیردومین با نام یک host در آن است که host یک سرویس یا کمپیوتر را تعیین می‌کند درحالی‌که زیردومین، دومین اصلی را وسعت می‌بخشد، که این خوبی را دارد که دومین اصلی خود را بهتر مدیریت کنیم.

۴.۳ طریقهٔ کار DNS

قبل از تشریح عملکرد سیستم DNS ضرور است تا با اصطلاحات مورد استفاده در این بحث آشنا شویم. ابتدا سرورهای نام (Name Server) را معرفی می‌کنیم که به کمپیوتری گفته می‌شود که در آن سرویس DNS فعال بوده و به‌عنوان جزئی از سیستم نام دامنه (Domain name system) فعالیت می‌کنند.

وظیفهٔ آنها تبدیل نام به IP و بالعکس می‌باشد. در داخل این سرورها فایل منطقه (Zone File) وجود دارد، این فایل یک فایل متنی ساده است که معلومات IP و نام یک دومین را در خود ذخیره می‌کند. مشابه فایل hosts ولی با ساختار متفاوت و معلومات بیشتر که توسط سرویس DNS مدیریت می‌شود. سرورهای نام می‌توانند مسئول (authoritative) باشند به این معنا که آنها جوابگوی تمام سوالات در مورد نام‌های دومینی که تحت کنترول آنها است، می‌باشند، در غیر این صورت سرور نام یا سرور کش (Cache server) است ویا تمام در خواست‌ها را به دیگر سرورها انتقال می‌دهد و خودش پاسخگو نیست.



شکل(۴-۵) ساختار درختی DNS تا دو سطح

وقتی که یک درخواست نام به یک سرور نام می‌رسد، سرور نام یکی از سه کار زیر را انجام می‌دهد:

- اگر جواب در حافظهٔ کش(Cache Memory) سرور باشد، پاسخ را از کش خود می‌دهد.
- اگر جواب در فایل منطقه (Zone file) باشد، پاسخ را از آن فایل می‌دهد، یک منطقه (zone) بخشی از ساختار درختی DNS است که در یک سرور نام ذخیره شده است. وقتی یک سرور نام میزبان[1] zone می‌باشد، در مقابل تمام نام‌های که در آن zone قراردارند پاسخگو می‌باشد؛ برای مثال: اگر یک سررور میزبان zone برای دومین contoso.com باشد، می‌تواند تمام درخواست‌های نام در contoso.com را جواب بدهد.
- اگر سرور نتواند از طریق کش یا فایل‌های zone به درخواست‌ها پاسخ دهد در این صورت از سایر سرورهای نام می‌پرسد.

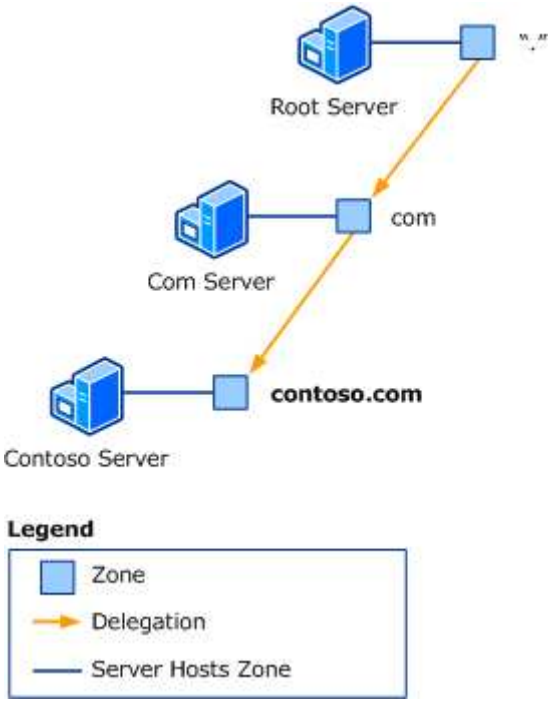
فهم و درک قابلیت‌های اصلی DNS مهم است، این قابلیت‌ها از قبیل تفویض (delegation)، تجزیه نام بازگشتی (recursive name resolution) است. که در ادامه آنها را بیشتر تشریح می‌کنیم.

۴.۳.۱ تفویض (delegation)

برای این‌که یک DNS سرور بتواند هر جستجوی نامی را جوابگو باشد، باید مسیر مستقیم یا غیر مستقیم به هر zone دیگر داشته باشد. این مسیرها توسط تفویض ایجاد می‌شوند. تفویض یک رکورد در parent zone است که سرور نامی را مشخص می‌سازد که مسئول نام‌ها در سطح بعدی ساختار درختی است. تفویض

[1] - host

این امکان را می‌دهد تا یک سرور نام در یک zone بتواند به کلاینت‌های یک سرور نام در zone دیگر اشاره کند. شکل زیر یک مثال از تفویض را نشان می‌دهد:



شکل (۴-۶) رابطهٔ تفویض یا Delegation بین سرورهای نام

همان‌طور که در شکل مشاهده می‌کنید، سرور ریشه DNS میزبان root zone است که با علامهٔ نقطه (.) در بالاترین سطح نشان داده شده است. Root zone یک تفویض به سطح بعدی یعنی come zone دارد. تفویض در root zone به سرور ریشه می‌گوید که برای IP پیداکردن come zone باید به سرور Com ارتباط بگیرد. به‌طور مشابه، تفویض در come zone به سرور Com می‌گوید، برای یافتن contoso.com zone باید با سرور Contoso ارتباط بگیرد.

نوت: تفویض دو نوع رکورد را استفاده می‌کند، NS رکورد که نام‌های سرورهای مسئول (authoritative server) را فراهم می‌سازد. همچنان رکوردهای میزبان (A) و میزبان (AAAA) که به ترتیبIPv4 و IPv6سرورهای مسئول را فراهم می‌سازد.

با استفاده از سلسلهٔ مراتب zoneها و تفویض (delegation)، یک سرور ریشه DNS می‌تواند هر نامی را پیدا کند زیرا در بالاترین سطح قرار دارد و با واگذاری آن به سرورهای دیگر درنهایت می‌تواند IP آن را که در یکی از سرورها در سطوح پایین‌تر قرار دارد، پیدا کند. هر سروری که بتواند از سرورهای ریشه نام‌ها را پرسان کند می‌تواند هر نامی را پیدا کند.

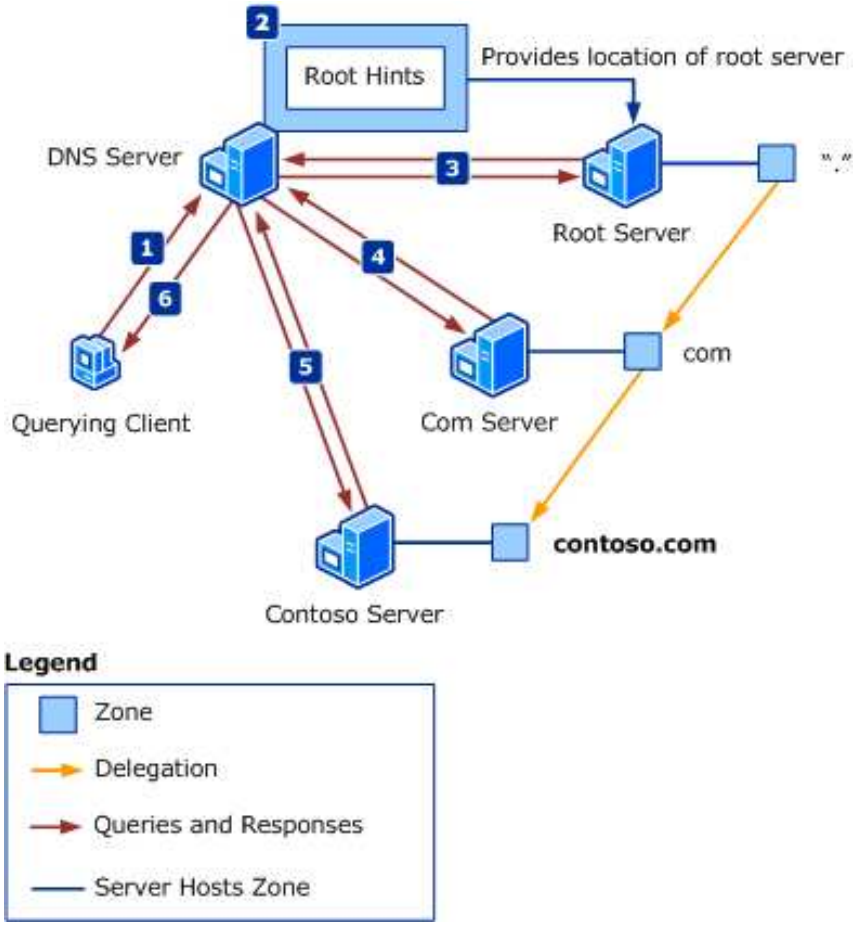
۴.۳.۲ تجزیه نام بازگشتی (Recursive name resolution)

پروسه‌یی است که سرور نام از ساختار سلسله‌مراتبی zoneها و تفویض استفاده می‌کند تا به سوالات در مورد نام‌های که مسئول آن نیست پاسخ دهد. اگر نام مربوط به خود سرور نام شود و سرور یک سرور مسئول باشد به آن پاسخ می‌دهد ولی اگر درخواست نام مربوط به خودش نباشد از تجزیهٔ نام بازگشتی برای یافتن پاسخ آن استفاده می‌کند.

برای تجزیه بازگشتی دو روش وجود دارند ویا به عبارت دیگر سرورهای نام به دو شکل عیارسازی می‌شوند، اول با استفاده از Root hints و دوم ارسال درخواست‌ها به سایر سرورها (forwarding)، که در ادامه تشریح می‌شوند.

۴.۳.۳ استفاده از root hints برای تجزیهٔ نام

سرورهای نام طوری عیارسازی می‌شوند که لست سروهای ریشه را در خود دارند به این لست (root hints) گفته می‌شود. این لست شامل نام وIP های سرورهای ریشه می‌باشد. همان‌طوری که قبلاً اشاره شد، هر سروری که به سرورهای ریشه دسترسی داشته باشد، می‌تواند تمام نام‌ها را پیدا کند.

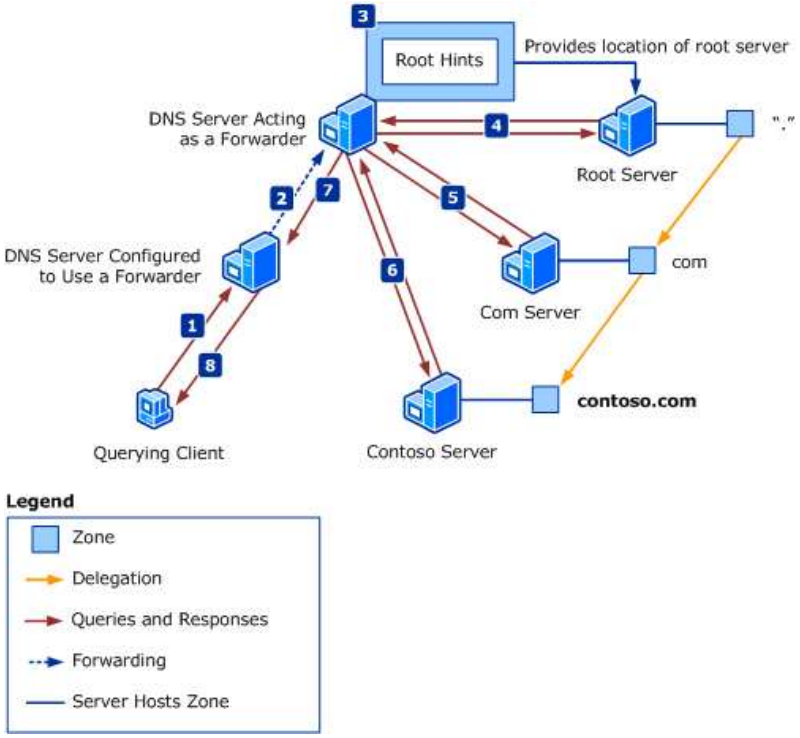


شکل(۴-۷) یافتن نام از روش root hint

- یک کلاینت یک درخواست به سرور DNS برای یافتن IP کمپیوتری به نام ftp.contoso.com می‌فرستد. کلاینت یک جواب قطعی و واضح در مورد آن می‌خواهد. جواب این درخواست یا یک IP معتبر است ویا یک پیغام که دلالت بر یافت‌نشدن آن، است.
- از آنجایی که سرور DNS مسئول آن نام نیست و جواب آن را نیز در کش یا ذخیره‌گاه خود ندارد، از طریق root hint که در خود ذخیره دارد آن را به یکی از سرورهای ریشهٔ نزدیک روان می‌کند.
- سرور DNS از درخواست های تکراری برای تجزیهٔ نام ftp.contoso.com استفاده می‌کند. درخواست‌های تکراری به این معنا است که سرور اجازه می‌دهد درخواست‌ها به سرورهای دیگر روان شود. چون آدرس ftp.contoso.com مربوط به دامنه com است، سرور ریشه یک ارجاع به سرور Com را بر می‌گرداند که com zone در آن قرار دارد.
- سرور DNS درخواست را به سرور Com روان می‌کند، از آنجایی که این نام به contoso.com ختم می‌شود، سرور Com یک ارجاع به سرور Contoso بر می‌گرداند که contoso.com zone در آن واقع است.
- سرور DNS به درخواست‌های تکراری خود ادامه می‌دهد و از سرور Contosso نام ftp.contoso.com را می‌پرسد. سرور Contoso مسئول این نام است و جواب را در Zone خود می‌یابد و جواب را به سرور DNS بر می‌گرداند.
- سرور IP را به کلاینت بر می‌گرداند.

۴.۳.۴ روش forwarding برای تجزیهٔ نام

روش دوم، فرستادن درخواست‌ها به سرورهای دیگر (forwarding) است. شکل زیر طریقهٔ یافتن نام از این روش را نشان می‌دهد:



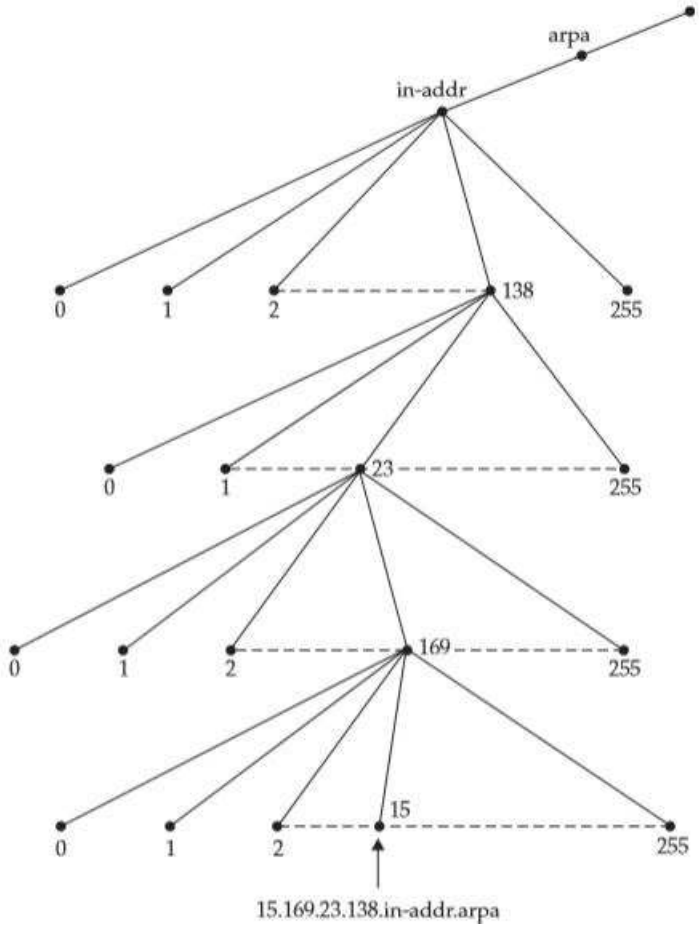
شکل(۴-۸) یافتن نام به روش forwarding

- کلاینت درخواست نام ftp.contoso.com را به سرور DNS اول روان می‌کند.
- سرور DNS اول درخواست را به سرور دوم که به نام forwarder است روان می‌کند.
- چون forwarder مسئول این نام نیست و جواب آن را نیز در کش یا ذخیره‌گاه خود ندارد. از Root hints برای یافتن IP آدرس یکی از سرورهای ریشه استفاده می‌کند.
- سرور forwarder از درخواست‌های تکراری برای یافتن نتیجه استفاده می‌کند که مراحل 4 تا 6 در شکل را شامل می‌شود.
- درنهایت IP یافت‌شده از سرور Contoso به سرور forwarder و از آنجا به DNS اول فرستاده شده تا به کلاینت داده شود.

۴.۳.۵ دومین in-addr.arpa

پروسهٔ تجزیه نام‌ها در دو جهت کار می‌کند. تجزیهٔ روبه‌جلو (forward resolution) که نام را به IP تجزیه می‌کند. و تجزیهٔ روبه‌عقب (backward resolution) که برعکس عمل می‌کند و IP را گرفته و نام را پیدا می‌کند. پروسهٔ تجزیه روبه‌عقب وظیفهٔ دومین in-addr.arpa است.

تجزیه روبه‌جلو از راست به چپ انجام می‌شد یعنی برای تجزیهٔ آدرس .support.mysite.com از سمت چپ شروع می‌کردیم یعنی علامهٔ نقطه که سرورهای ریشه بود، بعد از سرور com پرسان می‌کردیم و همین‌طور به سمت چپ ادامه می‌دادیم تا IP یافت می‌شد. اما در تجزیهٔ روبه‌عقب از IP را از چپ به راست نوشته و تجزیه را انجام می‌دهیم. برای تجزیه IP هم باید TLD یا دومین سطح بالا باید وجود داشته باشد که برای IPv4 دومین in-addr.arpa و برای IPv6 دومین ip6.arpa می‌باشد. شکل زیر یک مثال از تجزیهٔ آدرس 138.23.169.15 به نام را نشان می‌دهد.



شکل(۴-۹) ساختار in-addr.arpa برای تجزیهٔ IP

۴.۴ انواع سرورهای DNS

سرورهای DNS از نظر عیارسازی به‌طور عموم به سه دسته تقسیم‌بندی می‌شوند (Soyinka,۲۰۱۶):

- سرور DNS اصلی (primery DNS server) که به آنها سرور مسئول (authoritive server) نیز گفته می‌شود.
- سرور DNS ثانوی (secondery DNS server) که به آنها slave DNS server هم گفته می‌شود.
- سرور DNS کش (caching DNS server) که به آن resolver DNS server نیز گفته می‌شود.

یک سرور می‌تواند همزمان هم اصلی باشد و هم ثانوی، و تمام سرورها قابلیت کش‌سرور (cache server) بودن را نیز دارند. سرور اصلی همان سرور مسئول است که قبلاً تشریح داده شد، و مسئولیت پاسخ‌گویی به سوالات در مورد نام‌های مربوط به دومین خود را دارد. سرور اصلی همان سروری است که فایل‌های تنظیمات در آن وجود دارد. به بیان ساده‌تر، یک سرور اصلی، معلومات تمام hostها و زیردومین‌های تحت خود را دارد.

سرورهای ثانوی، به‌عنوان یک سرور پشتیبان و سرور توزیع load برای سرور اصلی DNS عمل می‌کند. این سرور یک کاپی از تمام معلومات سرور اصلی را در خود دارد. موجودیت سرورهای ثانوی در سرور اصلی تعریف می‌شود و سرور اصلی به‌طور دوره‌یی به‌روزرسانی تغییرات نام‌ها و IP ها را به سروهای ثانوی می‌فرستد تا به‌روز بمانند، تا درصورتی‌که سرور اصلی به هر دلیلی از کار بیفتد ویا دیگر قادر به پاسخ‌گویی به درخواست‌های تجزیه نام نباشد، این سرورهای ثانوی هستند که بار load اضافی را به دوش می‌کشند و به سرور اصلی کمک می‌کنند.

سرورهای کش، همان‌طور که از نامشان پیداست فقط به‌عنوان کش عمل می‌کنند و هیچ عیارسازی و تنظیماتی در مورد هیچ دومینی در آنها انجام نمی‌شود. این سرورها درخواست‌های بازگشتی را مدیریت می‌کنند و به‌طور معمول به منظور انجام جستجوی مغلق از سرورهای دیگر استفاده می‌شود. وقتی که یک کلاینت درخواست تجزیه نامی را به سرور کش می‌فرستد، این سرور ابتدا کش خود را چک می‌کند اگر جواب در کش موجود نباشد، درخواست را به سرور اصلی روان می‌کند و پس از گرفتن جواب آن را کش کرده و به کلاینت نیز برمی‌گرداند. معلومات کش‌شده را تا مدت معینی نگهداری می‌کند (به اندازهٔ TTL تعیین شده)، این سرورها سرعت جستجوهای نام را در شبکه افزایش می‌دهند.

۴.۵ دستورات مهم برای کار با DNS

تا این جای کار با مفهوم اولیه و طرز کار DNS آشنا شدید، وقت آن است تا کمی عملی‌تر کار کنیم. در این بخش با دستورات whois, host, dig آشنا می‌شویم که معلوماتی در مورد DNS به ما می‌دهد.

۴.۵.۱ دستور whois

شرکتی که شما از آن دومین را خریداری یا اجاره می‌کنید، ثبت‌کننده (registrar) نام دارد، شما می‌توانید سروهای نام برای دومین خود مشخص کنید، این سرورها معمولاً توسط ISP تعیین می‌شود، ISP ها این سرورهای نام را به ثبت‌کننده می‌دهند تا در TLDها ثبت کنند. شما از طریق دستور whois می‌توانید معلوماتی دیتابیس ثبت‌کننده‌ها را جستجو کنید و معلوماتی در مورد سروهای نام خود به دست آورید. البته برای استفاده از این دستور باید بستهٔ نرم‌افزاری whois را قبلاً نصب کرده باشید. به مثالی که در شکل زیر آمده است دقت نمایید:

```
$ whois 'domain google.com'
Whois Server Version 2.0

Domain names in the .com and .net domains can now be registered
with many different competing registrars. Go to http://www.internic.net
for detailed information.

   Domain Name: GOOGLE.COM
   Registrar: MARKMONITOR INC.
   Sponsoring Registrar IANA ID: 292
   Whois Server: whois.markmonitor.com
   Referral URL: http://www.markmonitor.com
   Name Server: NS1.GOOGLE.COM
   Name Server: NS2.GOOGLE.COM
   Name Server: NS3.GOOGLE.COM
   Name Server: NS4.GOOGLE.COM
   Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
   Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
   Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
   Status: serverDeleteProhibited https://icann.org/epp#serverDeleteProhibited
   Status: serverTransferProhibited https://icann.org/epp#serverTransferProhibited
   Status: serverUpdateProhibited https://icann.org/epp#serverUpdateProhibited
   Updated Date: 20-jul-2011
   Creation Date: 15-sep-1997
   Expiration Date: 14-sep-2020
```

شکل (۴-۱۰) مثالی از اجرای دستور whois برای یک دومین

در مثال فوق ما معلوماتی در مورد ثبت‌کننده و سروهای نام دومین google.com می‌خواهیم پیدا کنیم. معلوماتی از قبیل نام ثبت‌کننده، نام سرورهایی که دومین به آنها واگذار شده است و تاریخ ایجاد، تغییر و انقضا، نشان داده می‌شود.

۴.۵.۲ دستور host

برای دسترسی به نام‌هایی که در انترنت است شما به سرورهای DNS تعیین شده از طرف ISP نیاز دارید. آدرس این سرورهای DNS در فایل /etc/resolv.conf ذخیره شده است. زمانی‌که شما از طریق یک برنامه بخواهید نامی را بیابید، ابتدا فایل /etc/hosts جستجو شده و اگر یافت نشد به یکی از DNSها که در فایل /etc/resov.conf مشخص شده مراجعه می‌کند. این سرورها نیز به یکی از دو روشی که قبلاً تشریح شد عمل می‌کنند و نتیجه را یافته و به برنامهٔ مورد نظر تحویل می‌دهند.

```
$ cat /etc/resolv.conf
search example.com
nameserver 192.168.1.1
nameserver 192.168.1.254
```

شکل(۴-۱۱) محتوای یک فایل resolv.conf

شما می‌توانید از طریق دستور host در مورد این سرورهای DNS معلومات به دست آورید. البته قبلاً باید بستهٔ نرم‌افزاری bind-utils را نصب کرده باشید. این دستور جایگزین دستور قدیمی‌تر nslookup است که دیگر استفاده نمی‌شود. شما می‌توانید نام دومین خود را و همچنان آدرس IP سرور DNS را که می‌خواهید

از آن سوال کنید، تعیین کنید تا سرور DNS تمام IPهای مربوط به آن دومین را برای شما نشان دهد. به مثال شکل زیر توجه کنید:

```
$ host www.google.com 192.168.1.1
Using domain server:
Name: 192.168.1.1
Address: 192.168.1.1#53
Aliases:

www.google.com has address 150.101.161.167
www.google.com has address 150.101.161.173
www.google.com has address 150.101.161.174
www.google.com has address 150.101.161.180
www.google.com has address 150.101.161.181
www.google.com has address 150.101.161.187
www.google.com has address 150.101.161.146
www.google.com has address 150.101.161.152
www.google.com has address 150.101.161.153
www.google.com has address 150.101.161.159
www.google.com has address 150.101.161.160
www.google.com has address 150.101.161.166
www.google.com has IPv6 address 2404:6800:4006:800::2004
```

ما

شکل (۴۰-۱۲) مثالی از اجرای دستور host برای یافتن آدرس‌های دومین www.google.com

همچنان شما می‌توانید از به جای نام دومین از IP استفاده کنید تا به شما دومین را نشان دهد.

۴.۵.۳ دستور dig

این دستور مشابه دستور host است با این تفاوت که معلومات بیشتری را نشان می‌دهد. این دستور معلومات کافی برای برطرف‌کردن مشکلات DNS را در اختیار ما قرار می‌دهد. این دستور نیز در بستهٔ نرم‌افزاری bind-utils می‌باشد و همراه آن نصب می‌شود. به مثال زیر توجه کنید:

```
$ dig www.google.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33352
;; flags: qr rd ra; QUERY: 1, ANSWER: 12, AUTHORITY: 4, ADDITI

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         41      IN      A       150.101.161.153
www.google.com.         41      IN      A       150.101.161.159
www.google.com.         41      IN      A       150.101.161.160
www.google.com.         41      IN      A       150.101.161.166
www.google.com.         41      IN      A       150.101.161.167
```

```
;; AUTHORITY SECTION:
google.com.             2071    IN      NS      ns4.google.com.
google.com.             2071    IN      NS      ns1.google.com.
google.com.             2071    IN      NS      ns3.google.com.
google.com.             2071    IN      NS      ns2.google.com.

;; ADDITIONAL SECTION:
ns1.google.com.         179     IN      A       216.239.32.10
ns2.google.com.         4851    IN      A       216.239.34.10
ns3.google.com.         186     IN      A       216.239.36.10
ns4.google.com.         8300    IN      A       216.239.38.10

;; Query time: 11 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Sun Jun 26 00:11:48 UTC 2016
;; MSG SIZE rcvd: 371
```

بخش اول خروجی معلوماتی در مورد دستوری که اجرا کرده‌اید و وضعیت آن که آیا موفق اجرا شده است یا نه؟ نشان داده می‌شود و در بخش query section نشان می‌دهد که چه چیزی را شما واقعاً به سرور DNS فرستاده‌اید که به‌شکل یک A رکورد می‌باشد (انواع رکوردها بعدا تشریح می‌شود)، A رکورد نام را به IP وصل می‌کند. بخش بعدی answer section است که نتیجه یا جواب شما است و می‌گوید که چندین A رکورد برای دومین www.google.com وجود دارد. در بخش authority section لستی از سرورهای DNS مسئول را نشان می‌دهد. در بخش additional هم IP آدرس سرورهای DNS مسئول را نشان می‌دهد. جواب‌های واقعی در پنج ستون نشان داده می‌شود که بر اساس فارمت استندرد BIND[1] می‌باشد (Negus,۲۰۱۰). که هر خط یک رکورد است و هر رکورد از پنج بخش تشکیل شده و علامهٔ (;) برای تشریحات استفاده می‌شود. شکل زیر یک رکورد را نشان می‌دهد:

<record name>	<ttl>	<class>	<type>	<data>
www.google.com.	41	IN	A	150.101.161.153

شکل(۴-۱۳) شکل یک رکورد از معلومات DNS

اولین بخش، نام رکورد است (www.google.com)، بخش دوم ttl است که اشاره به مدت زمان اعتبار دیتا دارد، بخش سوم در انترنت همیشه IN است، بخش چهارم نوعیت رکورد است که انواع مختلفی دارد که بعدا تشریح می‌شود و بخش اخر هم دیتا یا همان آدرس IP است.

انواع فیلدها در جدول زیر تشریح شده است:

[1] - Berkeley Internet Name Domain

جدول(۴-۱) لست بعضی از نوعیت‌های رکوردهای DNS

Type		Used for
SOA	Start of Authority	تعریف سریانمبر و معلومات انقضا برای دومین
NS	Name Server	برای مشخص‌کردن DNS سرور برای دومین
A	Address Record	برای تعریف IPv4 آدرس برای نام host
AAA	Address Record	برای تعریف IPv6 آدرس برای نام host
PTR	Pointer Record	برای تعریف نام برای IP آدرس
MX	Mail Exchanger	تعیین سرور ایمل برای دومین
CNAME	Canonical Name	تعریف یک نام ثانوی یا نام مستعار برای یک رکورد A یا AAA

حال که شما با نوعیت مختلف رکوردهای DNS آشنا شدید می‌توانید از دستور dig بر اساس نوعیت استفاده کنید؛ برای مثال: اگر بخواهیم از صحت عملکرد یکی از DNSهای google که قبلاً از طریق دستور whois آنها را شناسایی کردیم، اطمینان حاصل نماییم؛ مثلاً: سرور ns1.google.com دستور زیر را می‌توان اجرا کرد:

```
$ dig @ns1.google.com google.com NS

; <<>> DiG 9.10.3-P4-Ubuntu <<>> google.com NS
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 44887
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;google.com.                    IN      NS

;; ANSWER SECTION:
google.com.             10158   IN      NS      ns2.google.com.
google.com.             10158   IN      NS      ns3.google.com.
google.com.             10158   IN      NS      ns1.google.com.
google.com.             10158   IN      NS      ns4.google.com.

;; ADDITIONAL SECTION:
ns1.google.com.         8267    IN      A       216.239.32.10
ns2.google.com.         12939   IN      A       216.239.34.10
ns3.google.com.         8274    IN      A       216.239.36.10
ns4.google.com.         1987    IN      A       216.239.38.10

;; Query time: 9 msec
;; SERVER: 10.0.2.3#53(10.0.2.3)
;; WHEN: Sun Jun 26 01:57:01 UTC 2016
;; MSG SIZE  rcvd: 175
```

شکل(۴-۱۴) مثال اجرای dig بر اساس نوعیت رکورد NS

و برای این‌که رابطه بین DNS سرورها را ببینیم می‌توانیم از dig +trace domain استفاده کنیم مطابق شکل زیر:

```
$ dig +trace www.google.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> +trace www.google.com
;; global options: +cmd
.                       9903    IN      NS      a.root-servers.net.
.                       9903    IN      NS      d.root-servers.net.
.                       9903    IN      NS      j.root-servers.net.
.                       9903    IN      NS      e.root-servers.net.
.                       9903    IN      NS      k.root-servers.net.
...<snip>...
.                       9903    IN      NS      h.root-servers.net.
...<snip>...
```

شکل(۴-۱۵) اجرای trace در یک دومین

۴.۶ عیارسازی DNS سرور لینوکس

بسته‌های نرم‌افزاری مختلفی برای راه‌اندازی قابلیت DNS در لینوکس وجود دارد ولی در این کتاب از نرم‌افزار معروف BIND استفاده می‌کنیم. اکثریت سرورهای لینوکس از این نرم‌افزار برای سرورهای DNS استفاده می‌کنند. در لینوکس‌های Redhat , centos شما می‌توانید از طریق دستور زیر آن را نصب کنید.

```
$ dnf –y install bind bind-utils
```

ویا از دستور yum هم می‌توانید استفاده کنید:

```
$ yum -y install bind*
```

سرویس DNS تحت نام named در لینوکس نصب می‌شود. بعد از نصب باید آن را start و فعال (enable) کنید.

```
$ systemctl start named
$ systemctl enable named
```

۴.۶.۱ تنظیمات BIND

فایل /etc/named.conf فایل اصلی تنظیمات BIND است. تنظیمات DNS سرور در این فایل صورت می‌گیرد. برای تنظیم‌کردن DNS سرور ابتدا باید ساختار این فایل را بفهمیم. فارمت عمومی تنظیمات این فایل به شکل زیر است:

```
statement {
        options; // comments
};
```

در آخر هر options از علامهٔ ; استفاده می‌شود. دستورات (statment) مختلفی وجود دارد، که در جدول زیر مهمترین آنها آمده است:

جدول(۴-۲) بعضی از دستورات BIND

statments	Description
options	برای تنظیمات سراسری BIND استفاده می‌شود.
logging	برای مشخص‌کردن مواردی که باید log از آنها گرفته شود یا نشود استفاده می‌شود و همچنان محل ذخیره‌کردن logها استفاده می‌شود.
Server	برای تعریف تنظیمات مخصوص سرور DNS از آن استفاده می‌شود.
zone	برای تعریف DNS zone استفاده می‌شود.
include	برای شامل‌کردن فایل دیگر در فایل named.conf استفاده می‌شود.

فولدر کار BIND در دستور options تعریف می‌شود که به‌طور پیش‌فرض /var/named می‌باشد ولی شما می‌توانید آن را تغییر دهید.

```
options{
directory "/var/named"; // put config files in /var/named
;{
```

از علامه‌های //,/* ... */ , # نیز برای نوشتن، کامنت و تشریحات می‌توان استفاده کرد که در جملهٔ تنظیمات به حساب نمی‌آید. مسیر پیش‌فرض برای کار BIND آدرس /var/named می‌باشد. Logها به‌طور پیش‌فرض در آدرس /var/log/messages ذخیره می‌شوند. تنظیمات پیش‌فرض logging برای اکثر کاربران کافی است و ضرور به تغییر آن نیست، این تنظمات در شکل زیر آمده است:

```
1.  logging {
2. category default { default_syslog; };
3. category queries { default_syslog; };
4.
5. };
```

شکل (۴-۱۶) تنظیمات پیش‌فرض logging

از دستور server نیز برای معرفی سرورهای نام دیگری که ارتباط دارد استفاده می‌شود، برای مثال در شکل زیر ما یک سرور با آدرس ۱۹۲.۱۶۸.۱.۱۲ را معرفی می‌کنیم:

```
server 192.168.1.12 {
bogus no;
transfer-format many-answers;
};
```

شکل ۰(۴-۱۷) معرفی یک سرور

Zone از نظر تیوری زیرمجموعه‌یی از معلومات یک دومین گفته می‌شود. یک دومین می‌تواند شامل زیردومین نیز باشد در این صورت هر زیردومین یک zone خواهد داشت. تمام zoneها با هم یک دومین را تشکیل می‌دهند، zoneها شامل رکوردهایی هستند که معلومات نام و IP دومین را در خود ذخیره می‌کنند. وقتی یک zone را در فایل /etc/named.conf تعریف می‌کنید، باید فایل دیتابیس آن را نیز مشخص بسازید معمولاً نام فایل همان نام Zone با پسوند .db می‌باشد برای مثال اگر یک zone برای دومین example.org تعریف کنیم در این صورت نام فایل می‌تواند example.org.db باشد، این فایل در مسیر پیش‌فرض یعنی /etc/named/example.org.db ذخیره می‌شود؛ برای مثال ما می‌خواهیم یک سرور DNS اصلی برای دومین example.org تعریف کنیم، برای این کار در فایل named.conf یک zone اصلی باید تعریف کنیم:

```
zone "example.org" {
 type master;
 file "example.org.db";
};
```

این zone یک forward lookup zone است یعنی از طریق آن می‌توان نام سایت را جستجو کرد و IP را یافت. باید یک backward lookup zone نیز تعریف شود تا عمل عکس انجام شود. برای تعریف zone برعکس باید IP آدرس سرور اصلی را به‌صورت بر عکس نوشته و بعد از آن in-addr.arpa را می‌نویسیم. به کود زیر توجه کنید:

```
zone "1.168.192.in-addr.arpa" {
        type master;
        file "example.org.rev";
};
```

دقت کنید که نام فایل‌ها اختیاری است و شما هر نام دیگر را نیز می‌توانید اختیار کنید.

۴.۶.۲ تنظیمات فایل دیتابیس zone ها

همان‌طور که قبلاً گفته شد برای هر Zone که تعریف می‌شود یک فایل دیتابیس در نظر گرفته می‌شود، این فایل دیتابیس در اصل یک فایل متنی ساده با فارمت خاص است که در این بخش می‌خواهیم آن را بیشتر تشریح کنیم. کامنت‌ها یا تشریحات در این فایل با ; مشخص می‌شود. هر فایل دیتابیس باید با $TTL شروع شود که مدت اعتبار معلومات رکوردها را مشخص می‌سازد. بعد از آن انواع رکوردهای دیگر تعریف می‌شود. ساختار یک فایل دیتابیس Zone به شکل زیر است:

```
$TTL
SOA  record
NS   records
MX   records
A  and  CNAME records
```

شکل(۴-۱۸) ساختار یک فایل دیتابیس Zone

۴.۷ انواع رکوردهای DNS

همان‌طور که در شکل فوق می‌بینید انواع رکوردها باید در فایل دیتابیس تعریف شود که برای تنظیم DNS باید با هر یک از آنها آشنا شویم.

۴.۷.۱ رکوردهای SOA

رکورد SOA به معنای «آغاز اعتبار» (Start of Authority) یک رکورد اجباری در همه فایل‌های منطقه است. این مورد باید نخستین رکورد واقعی در یک فایل منطقه باشد (گرچه ممکن است مقادیر $ORIGIN یا $TTL بالاتر از آن واقع شده باشند.) رکورد آغاز اعتبار مانند زیر است:

```
$TTL 604800
example.com. IN SOA ns1.example.com. mail.example.com).
2017012604     ;serial
86400   ;refresh, seconds
7200            ;retry, seconds
3600000         ;expire, seconds
86400   ;minimum, seconds
(
```

در سطر اول TTL آمده است، این قیمت که بر اساس ثانیه تعیین می‌شود به سرور DNS مدت زمان اعتبار رکوردها را یادآوری می‌کند؛ برای مثال: اگر قیمت آن ۱۴۴۰۰ ثانیه باشد که برابر ۴ ساعت است، سروهای

DNS تا ۴ ساعت zoneهای شما را کش (cache) می‌کنند و بعد از ختم این وقت دوباره از سرور شما در مورد نام‌ها سوال می‌کند (Soyinka,۲۰۱۶).

در سطر دوم که شروع SOA است ابتدا نام دومین آمده و IN نشانهٔ internet است و بعد از آن SOA که نوعیت رکورد است و سپس نام سرور DNS می‌آید که در اینجا .ns1.example.com است، این سرور مسئول دومین example.com است، در آخر هم آدرس ایمل مدیر دومین می‌آید که البته به جای علامهٔ @، علامه . آمده است یعنی آدرس ایمل .mail@example.com مد نظر بوده است. معلومات بعدی که در در بین قوس‌ها آمده است، رفتار DNS سرور را تعیین می‌کند.

Serial: این شماره سریال فایل منطقه است. زمانی‌که یک فایل منطقه را ویرایش می‌کنید، در اصل می‌توان گفت شمارهٔ نسخهٔ فایل است، اگر این نسخه اضافه شود یعنی فایل به‌روزرسانی شده است و سرورهای دیگر از جمله سرورهای ثانوی از این عدد می‌فهمند که سرور اصلی رکوردهای خود را به‌روزرسانی کرده و باید به‌روزرسانی جدید را از سرور اصلی بگیرند.

Refresh: مشخص می‌کند که سرور ثانوی هر چند ثانیه به سرور اصلی جهت چک‌کردن به‌روزرسانی‌های جدید سر بزند.

retry : در سطر پنجم، مدت زمان انتظار سرور ثانوی برای اتصال به سرور اصلی است درصورتی‌که نتواند به سرور اصلی متصل شود.

Expire: اگر سرور ثانوی نتواند به سرور اصلی متصل شود، مدت زمانی‌که معلوماتش معتبر خواهد بود در این بخش تعیین می‌شود.

Minimum: هم زمان انتظار برای کش سرورهایی را که نمی‌توانند به سرور اصلی متصل شوند نشان می‌دهد.

۴.۷.۲ رکوردهای NS

این رکوردها برای تعیین سرورهایی است که رکوردهای zone فعلی را قرار است در خود نگهداری کنند. معمول است که حد اقل دو سرور DNS باید تعیین شود یکی اصلی و دیگری ثانوی می‌باشد. فارمت آن به شکل زیر است:

```
IN NS ns1.example.com.
IN NS ns2.example.com.
```

۴.۷.۳ رکورد MX

این رکوردها برای معرفی سروهای ایمل داخلی استفاده می‌شود. فارمت آن به شکل زیر است:

```
IN MX 10 smtp1
IN MX 20 smtp2
```

اعداد ۱۰ و ۲۰ اولویت یا اهمیت سرور ایمل را نشان می‌دهد، هر چقدر کمتر باشد اهمیت آن بیشتر است.

۴.۷.۴ رکوردهای A و AAAA

هر دوی این رکوردها یک میزبان را به آدرس IP مربوطه مرتبط می‌سازند. رکورد A برای ارتباط یک میزبان به آدرس‌های IPv4 استفاده می‌شود؛ درحالی‌که رکوردهای AAAA برای ارتباط میزبان‌ها به آدرس‌های IPv6 استفاده می‌شوند. فارمت کلی این رکوردها به‌صورت زیر است:

```
ns1     IN A 192.168.0.100
client1 IN AAAA fe80::192.168.1.105
www     IN   A   192.168.0.80
@       IN   A   192.168.0.100
*       IN   A   192.168.0.100
```

قبل از IN باید نام یک میزبان را تعیین کنیم، از علامت @ می‌توان به جای نام دومین استفاده کرد و از علامت * می‌توان به جای تمام نام‌های نامشخص در این دومین استفاده کرد.

۴.۷.۵ رکوردهای CNAME

این رکورد برای تعریف نام مستعار یا نام ثانوی برای یک میزبان استفاده می‌شود؛ برای مثال: یک سرور به نام server۱ داریم و می‌خواهیم که یک نام مستعار (مثلاً www) به آن بدهیم تا با هر دو نام بتوانیم به آن سرور دسترسی داشته باشیم، باید به شکل زیر عمل کنیم:

```
server1 IN A      111.111.111.111
www     IN CNAME  server1
```

۴.۷.۶ رکوردهای PTR

رکوردهای PTR عکس پروسهٔ تجزیهٔ نام را انجام می‌دهند یعنی شما یک آدرس IP می‌دهید و این رکورد یک نام برای شما نتیجه می‌دهد. رکوردهای PTR عکس رکورد A یا AAAA هستند. رکوردهای PTR منحصربه‌فرد (unique) هستند یعنی تکراری تعریف نمی‌شوند. نمون‌های از این رکورد در زیر آمده است.

```
192.168.1.5 IN PTR support.example.com.
```

۴.۸ یک سناریوی واقعی شبکه

در این بخش یک سناریوی واقعی شبکه را مرحله به مرحله اجرا می‌کنیم تا از تمام مطالب گفته‌شده در فوق به‌طور عملی استفاده نمایید. هدف این سناریو عیارسازی انواع سروهایDNS اعم از primery, secondery , cashe می‌باشد. یک شبکه با دو سرور DNS و یک کلاینت در نظر می‌گیریم. مشخصات این سیستم‌ها در جدول زیر آمده است:

جدول(۴-۳) مشخصات نودهای شبکه

Server/client	Details
Primary (Master) DNS Server	Operating System : CentOS 7 minimal server Hostname : ns1.example.com IP Address : 192.168.1.101/24
Primary (Master) DNS Server	Operating System : CentOS 7 minimal server Hostname : ns2.example.com IP Address : 192.168.1.102/24
Client	Operating System : Windows 7 Hostname : pc1.example.com IP Address : 192.168.1.103/24
Web server	Operating System : CentOS 7 Web server Hostname : www.example.com IP Address : 192.168.1.104/24
Mail server	Operating System : CentOS 7 smtp server Hostname : mail.example.com IP Address : 192.168.1.105/24

قبل از هر چیز باید به سیستم‌های خود نام بدهیم. برای این کار باید در هر یک از سیستم‌ها فایل etc/hosts/ را باز کرده و IP و نام آنها را وارد کنیم برای مثال برای سرور ns۱ باید فایل به شکل زیر شود:

```
127.0.0.1       localhost
192.0.2.1       ns1.example.com      ns1
```

خط اول از قبل وجود دارد فقط کافی است خط دوم را اضافه کنیم. همچنان باید نام را بدون دومین در فایل /etc/hostname نیز اضافه کنیم؛ برای مثال: برای سرور ns۱ فایل /etc/hostname باید به شکل زیر شود:

```
$ cat /etc/hostname
ns1
$ hostname –F /etc/hostname
```

دستور آخر برای آن است که تغییرات اعمال شود. این دو کار را باید برای تمام سرورها و کلاینت‌ها باید تکرار کنیم و نام‌ها و IP هر یک را مطابق جدول فوق تنظیم کنیم.

۴.۹ عیارسازی سرور DNS اصلی

سرور ns1.example.com، به‌عنوان سرور DNS اصلی (primery DNS server) است. ابتدا سرویس bind را روی آن نصب می‌کنیم (طریقهٔ نصب و فعال‌سازی این سرویس قبلاً در همین فصل تشریح شده است.

۴.۹.۱ تنظیمات سرور DNS

فایل /etc/named.conf را با یک ادیتور مثل vi باز می‌کنیم و خطوطی که بولد (bold) شده‌اند را تغییر می‌دهیم. همچنان سه zone را نیز در انتهای فایل اضافه می‌کنیم.

```
options {
 listen-on port 53 { 127.0.0.1; 192.168.1.101;};### Master DNS IP ###
# listen-on-v6 port 53 {::1; };
4- directory "/var/named";
 dump-file "/var/named/data/cache_dump.db";
 statistics-file "/var/named/data/named_stats.txt";
 memstatistics-file "/var/named/data/named_mem_stats.txt";
8- allow-query { localhost; 192.168.1.0/24;}; ### IP Range ###
9- allow-transfer{ localhost; 192.168.1.102; }; ### Slave DNS IP ###
10- recursion yes;
 dnssec-enable yes;
 dnssec-validation yes;
 dnssec-lookaside auto;
 bindkeys-file "/etc/named.iscdlv.key";
 managed-keys-directory "/var/named/dynamic";
 pid-file "/run/named/named.pid";
 session-keyfile "/run/named/session.key";
};
logging {
 channel default_debug {
 file "data/named.run";
 severity dynamic;
 };
};
zone "." IN {
 type hint;
 file "named.ca";
};
zone "example.com" IN {
        type master;
        file "forward.example.db";
```

در سطر 8 رنج IP آدرس‌هایی را که اجازهٔ کیوری‌گرفتن از این سرور دارند، مشخص کردیم. در سطر 9 آدرس سرور ثانوی را برای انتقال رکوردهای DNS و به‌روزرسانی‌های آن مشخص شده است که در این سناریو 192.168.1.102 می‌باشد (همان سرور ns2.example.com). در سطر 10 یعنی این سرور در مورد نام‌های که مربوط به خودش نیست اجازه دارد که از سرورهای دیگر، مثل سروهای ریشه بپرسد و از روش تجزیهٔ نام بازگشتی برای یافتن IP آنها استفاده کند.

در آخر هم سه Zone تعریف شده است، zone اول باعث می‌شود که این سرور به یک سرور کش (cache server) نیز تبدیل شود و آدرس root hintها در فایل name.ca ذخیره شده است و نام‌هایی که مربوط به دومین خودش نیست را از این سروها به‌صورت بازگشتی می‌پرسد و در خود کش می‌کند. Zone دوم یک forward zone است و سومی نیز backward zone است.

۴.۹.۲ ایجاد فایل دیتابیس zoneها

برای هر یک از Zone های forward , backward باید یک فایل با نام‌های تعیین‌شدهٔ (reverse.example.db , forward.example.db) باید ایجاد کرد. این فایل‌ها باید در آدرس /var/named ایجاد شوند. در فایل forward.example.db کُدهای زیر را اضافه کنید:

```
$TTL 86400
@ IN SOA ns1.example.com. root.example.com. (
 2011071001    ;Serial
 3H     ;Refresh (3 hours)
 30M    ;Retry  (30 minutes)
 2W     ;Expire (2 weeks)
 1W     ;Minimum TTL (1 week)
)
@ IN NS ns1.example.com.   ;primary name server
@ IN NS ns2.example.com.   ;secondary name server
@ IN MX 10 mail.example.com.       ;mail server
ns1     IN A 192.168.1.101
ns2     IN A 192.168.1.102
pc1     IN A 192.168.1.103
www   IN A 192.168.1.104
mail    IN A 192.168.1.105
```

نوت: با توجه به معرفی هر یک از انواع رکوردها در مراحل قبل، دیگر از تشریحات اضافه در این بخش خودداری می‌کنیم. اگر بخش‌هایی از کُدهای فوق قابل فهم نیست به مطالب قبل رجوع نمایید.

در فایل reverse.example.db کُدهای زیر را اضافه کنید:

```
$TTL 86400
@ IN SOA ns1.example.com. root.example.com. (
 2011071001    ;Serial
 3H     ;Refresh (3 hours)
 30M    ;Retry  (30 minutes)
 2W     ;Expire (2 weeks)
 1W     ;Minimum TTL (1 week)
)
@ IN NS ns1.example.com.   ;primary name server
@ IN NS ns2.example.com.   ;secondary name server
@       IN PTR              example.com.
@ IN MX 10 mail.example.com.       ;mail server
ns1     IN A 192.168.1.101
ns2     IN A 192.168.1.102
pc1     IN A 192.168.1.103
www     IN A 192.168.1.104
mail    IN A 192.168.1.105
101     IN PTR              ns1.example.com.
102     IN PTR              ns2.example.com.
103     IN PTR              pc1.example.com.
104     IN PTR              www.example.com.
105     IN PTR              mail.example.com.
```

۴.۹.۳ فعال‌کردن سرویس DNS

با کُدهای زیر سرویس BIND را فعال کنید:

```
systemctl enable named
systemctl start named
```

۴.۹.۴ تنظیمات firewall

سرویس BIND به‌طور پیش‌فرض از پورت ۵۳ استفاده می‌کند، این پورت باید در تنظیمات firewall باید اضافه شود برای این کار از کُدهای زیر استفاده کنید:

```
firewall-cmd --permanent --add-port=53/tcp
firewall-cmd --permanent --add-port=53/udp
firewall-cmd –reload
```

۴.۹.۵ تنظیمات حق دسترسی و مالکیت سرویس

برای کار با سرویس BIND باید عضو گروپ named باشید این گروپ بعد از نصب این سرویس به‌طور خودکار در سیستم ایجاد می‌شود که مجوزهای لازم برای آن تعریف شده است. برای تنظیم حق دسترسی این سرویس کافی است که مسیر /var/named را به این گروپ «ست» کنیم و همچنان مالکیت فایل تنظیمات عمومی آن را به کاربر root می‌دهیم. این کارها با اجرای کُدهای زیر امکان‌پذیر است:

```
chgrp named -R /var/named
chown -v root:named /etc/named.conf
```

۴.۹.۶ چک تنظیمات

از کجا باید فهمید که تنظیمات انجام‌شده تا این مرحله صحیح است یا خیر؟ خوشبختانه برای چک‌کردن تنظیمات ابزارهایی وجود دارد، برای چک‌کردن تنظیمات در فایل /etc/named.conf از دستور named-checkconf و برای چک‌کردن تنظیمات zoneها از دستور named-checkzone استفاده می‌کنیم. به کُدهای زیر دقت کنید:

```
named-checkconf /etc/named.conf
named-checkzone example.com /var/named/forward.example.db
named-checkzone example.com /var/named/reverse.example.db
```

اگر خروجی این دستورات OK باشد، یعنی مشکلی نیست و تنظیمات درست انجام شده است.

۴.۹.۷ تنظیمات نهایی

IP سرور DNS اصلی را باید در انترفیس شبکه اضافه کنید، برای این کار باید "DNS="192.168.1.101 در فایل /etc/sysconfig/network-scripts/ifcfg-enp0s3 اضافه کنید. البته نام به جای ifcfg-enp0s3 ممکن است نام دیگری در سیستم شما باشد.

همچنان آدرس سرور DNS اصلی را باید در فایل /etc/resolv.conf نیز اضافه کنید. کُد زیر را باید اضافه شود:

```
nameserver 192.168.1.101

nameserver 192.168.1.101
```

برای این‌که تغییرات اعمال شود یک بار از طریق دستور زیر سرویس یا خدمات شبکه را restart کنید:

```
systemctl restart network
```

۴.۹.۸ امتحان‌کردن سرور DNS

حال وقت آن رسیده است تا سرور را امتحان کنیم که آیا صحیح کار می‌کند یا خیر؟ برای این کار از دستورات host و dig که قبلاً شرح داده شد استفاده کنید.

```
host example.com
dig ns1.example.com
```

۴.۱۰ عیارسازی سرور DNS ثانوی

ابتدا سرویس BIND را نصب می‌کنیم. سپس فایل /etc/named.conf را باز کنید و بخش‌های بولد (bold) شده در کُدهای زیر را به آن اضافه کنید:

```
options {
listen-on port 53 { 127.0.0.1; 192.168.1.102; };
listen-on-v6 port 53 {::1; };
directory "/var/named";
dump-file "/var/named/data/cache_dump.db";
 statistics-file "/var/named/data/named_stats.txt";
 memstatistics-file "/var/named/data/named_mem_stats.txt";
allow-query { localhost; 192.168.1.0/24; };
.
.
.
.
zone "." IN {
        type hint;
        file "named.ca";
};
```

```
zone "example.local" IN {
        type slave;
        file "slaves/example.fwd";
        masters { 192.168.1.101; };
};
zone "1.168.192.in-addr.arpa" IN {
        type slave;
        file "slaves/ example.rev";
        masters { 192.168.1.101; };
};
include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
```

را فعال کنید:BIND سرویس

```
systemctl enable named
systemctl start named
```

بعد از اجرای کُد فوق، به‌طور خودکار رکوردهای zoneها از سرور اصلی در فایل دیتابیس ,example.fwd example.rev انتقال می‌یابند. این فایل‌ها در آدرس /var/named/slaves موقعیت دارند می‌توانید با دستور ls آنها را مشاهده کنید.

باقی تنظیمات سرور ثانوی مشابه سرور اصلی است، که در مرحلهٔ قبل تشریح شد، تنظیمات firewall و تنظیمات حق دسترسی و مالکیت را انجام دهید. و برای اطمینان تمام تنظیمات را از طریق دستورات named-checkconf , named-checkzone یکبار چک کنید.

IP سرور DNS ثانوی و اصلی را باید در انترفیس شبکه اضافه کنید، برای این کار باید DNS=”192.168.1.101” و DNS=”192.168.1.102”

را در فایل ifcfg-enp0s3 اضافه کنید/etc/sysconfig/network-scripts/ifcfg-enp0s3. البته به جای ممکن است نام دیگری در سیستم شما باشد.

همچنان آدرس سرور DNS اصلی و سرور فعلی (ثانوی) را باید در فایل /etc/resolv.conf نیز اضافه کنید. کُد زیر را باید اضافه شود:

```
nameserver 192.168.1.101
nameserver 192.168.1.102
systemctl restart network
```

از طریق دستورات host و dig که قبلاً شرح داده شد استفاده کنید و سرور را متحان کنید:

```
host example.com
dig ns1.example.com
```

۴.۱۰.۱ تنظیمات کلاینت

در تمام کلاینت‌ها باید آدرس سرورهای DNS اصلی و ثانوی را در فایل /etc/resolv.conf ثبت کنید. و فایل باید به شکل زیر شود:

```
search example.com
nameserver 192.168.1.101
nameserver 192.168.1.102
```

خلاصهٔ فصل چهارم

- در سیستم‌عامل لینوکس فایل /etc/hosts برای ذخیره و ثبت نام و IP کمپیوترهای شبکه محلی استفاده می‌شود.
- سایت‌ها از طریق آدرس انترنتی آنها که به آن آدرس FQDN[1] نیز گفته می‌شود، باز می‌شوند.
- سه سطح دومین وجود دارد به نام‌های دومین ریشه، دومین سطح بالا، دومین سطح دوم.
- دومین‌های ریشهٔ آدرس سرورهای root را دارند.
- وظیفهٔ سرور DNS تبدیل نام به IP و بالعکس می‌باشد. در داخل این سرورها فایل منطقه (Zone File) وجود دارد، این فایل یک فایل متنی ساده است که معلومات IP و نام یک دومین را در خود ذخیره می‌کند.
- تجزیه نام بازگشتی، پروسه‌یی است که سرور نام از ساختار سلسله‌مراتبی zoneها و تفویض استفاده می‌کند تا به سوالات در مورد نام‌های که مسئول آن نیست پاسخ دهد.
- تجزیهٔ نام بازگشتی از دو روش استفاده می‌کند، یک روش استفاده از root hint است و روش دوم forwarding می‌باشد.
- تجزیهٔ روبه‌عقب (backward resolution) که برعکس عمل می‌کند و IP را گرفته و نام را پیدا می‌کند. پروسهٔ تجزیه روبه‌عقب بر شانه‌های دومین in-addr.arpa است.
- انواع سروهای DNS سه نوع هستند، سرور DNS اصلی (primery DNS server) که به آنها سرور مسئول (authoritive server) نیز گفته می‌شود، سرور DNS ثانوی (secondery DNS server) که به آنها slave DNS server هم گفته می‌شود و سرور DNS کش (caching DNS server) که به آن resolver DNS server نیز گفته می‌شود.
- دستورات whois, host, dig برای چک‌کردن و امتحان‌کردن تنظیمات سرور DNS و اطمینان از صحت کار آن استفاده می‌شود.
- سرویس BIND یک نرم‌افزار جهت عیارسازی سرور DNS در لینوکس است.
- برای عیارسازی یک سرور DNS در لینوکس با استفاده از BIND، بعد از نصب BIND باید فایل تنظیمات /etc/named.conf را باز کرده و Zoneهای مختلف را در آن تعریف نماییم. سپس هر یک از فایل‌های zone را باز کرده و رکوردهای A,AAA, MX, NS , ... را در آن تعریف می‌کنیم.

[1]- - fully qualified domain name

سوالات و فعالیت های فصل چهارم

۱. ساختار فایل /etc/hosts را تشریح کنید.
۲. فرق بین ساختار سیستم‌فایل و ساختار DNS را بیان کنید.
۳. ساختار و نحوهٔ کار سرور DNS را تشریح نمایید.
۴. انواع DNS سرور و نقش هر یک را بیان کنید.
۵. سطوح مختلف دومین را نام برده و مثال بزنید.
۶. Root hint چیست و چه کاربردی دارند؟
۷. دومین in-addr.arpa چه وظیفه‌یی دارد و چگونه تعریف می‌شود؟
۸. انواع رکوردهای DNS را نام برده و تشریح کنید.

فعالیت ها

۱. یک دومین به نام school.af را در نظر بگیرید و تحت این دومین دو سرور DNS، یک سرور ایمل، یک وب‌سرور و یک کمپیوتر معمولی با نام‌های مختلف و دلخواه در نظر بگیرید و مانند سناریوی جدول شماره پنج این فصل، یک جدول ترتیب دهید.
۲. به هر یک از کمپیوترها یک نام بدهید. این کار را در فایل /etc/hosts و /etc/hostname انجام دهید.
۳. نرم‌افزار BIND را روی سرور اصلی DNS خود نصب کنید و آن را مطابق توضیحات داده‌شده در آخر این فصل عیار بسازید.
۴. Zoneها و رکوردها را در فایل‌های دیتابیس zoneها تعریف کنید.
۵. Firewall را برای سرور اصلی DNS تنظیم کنید.
۶. تنظیمات سرور را با استفاده از دستورهای named-checkconf و... بررسی کنید.
۷. سرور را با استفاده از دستورات host,dig بررسی کنید.
۸. از هر یک از کمپیوترها با استفاده از دستور host, dig از صحت کار سرور DNS اصلی اطمینان حاصل نمایید.
۹. سرور ثانوی DNS را نیز به همان شکل عیار بسازید.

فصل پنجم

فایل‌سرور (File Server)

هدف کلی: در مورد تنظیم فایل‌سرور (File Server) در لینوکس شناخت حاصل نمایند.

اهداف آموزشی: در پایان این فصل محصلان قادر خواهند شد تا:

۱. فایل‌سرور و انواع پروتوکول‌های اشتراک فایل در لینوکس را تشریح کنند.
۲. نحوۀ کار و عیارسازی سرور Samba، تشریح کنند.
۳. نحوۀ کار و عیارسازی فایل‌سرور NFS را تشریح کنند.

۵.۱ فایل‌سرور (file server)

فایل‌سرور به کمپیوتری در شبکه گفته می‌شود که فایل‌های شبکه را به‌طور متمرکز با قابلیت اطمینان بالا ذخیره می‌کند و آن را برای کلاینت‌های شبکه به اشتراک می‌گذارد. برای راه‌اندازی یک فایل‌سرور جنبه‌های مختلفی باید در نظرگرفته شود، تکنالوژی‌های ذخیره‌سازی مثل NAS, SAN، دیزاین شبکه و پروتوکول‌های اشتراک‌گذاری فایل از مهمترین این جنبه‌ها است. از آنجایی که تعداد زیادی از کاربران همزمان به فایل‌سرور مراجعه می‌کنند و بار (Load) زیادی بر روی این سرورها می‌باشد لذا سخت‌افزار ذخیره‌سازی که برای این سرورها استفاده می‌شوند، باید متفاوت از سیستم‌های معمولی باشد، این سرورها باید قابلیت پاسخ‌گویی به تمام درخواست‌های فایل را در زمان کم را داشته باشند لذا باید از تکنالوژی‌هایی مثل SAN, NAS استفاده کنند. از نظر دیزاین شبکه نیز معمولاً از چندین سرور برای اشتراک فایل‌ها استفاده می‌شود و سرورها Cluster می‌شوند تا لود به‌طور مساویانه بین آنها تقسیم شود. بررسی این موضوعات خارج از موضوع کتاب می‌باشد و در این فصل فقط به بررسی پروتوکول‌های اشتراک‌گذاری فایل می‌پردازیم.

از مهمترین پروتوکول‌های اشتراک‌گذاری فایل FTP, NFS و Samba است.

۵.۲ انواع فایل سرورها

FTP مخفف File Transfer Protocol است و روشی عمومی برای انتقال فایل در شبکه مورد استفاده قرار می‌گیرد. شما می‌توانید با استفاده از FTP دانلود و آپلود فایل داشته باشید. و این بیشتر برای انتقال فایل بین یک کلاینت و سرور یا وب‌سایت کاربرد دارد. شما با استفاده از port forwarding می‌توانید به FTP سرور خود از دنیای انترنت هم دسترسی داشته باشید. برنامه‌های بسیاری از جمله filezilla یا cuteftp و... استفاده می‌کنند. عیارسازی این نوع فایل‌سرور در فصل بعدی معرفی می‌شود.

پروتوکول SMB مخفف عبارت Server Message block است و یک پروتوکول application layer protocol است. این پروتوکول است که عمدتاً برای دسترسی به پرنترها، فایل‌ها و پورت‌ها استفاده می‌شود ضمن این‌که این پروتوکول قابلیت تصدیق هویت (authentication) کاربران را نیز دارد. SAMBA پیاده‌سازی‌یی از SMB است و CIFs هم پیاده‌سازی خاص از SMB است و مخفف عبارت Common Internet File System است.

NFS مخفف Network File System است و یک پروتوکول استندرد است که در یک فایل‌سیستم توزیع‌شده (distributed file system) استفاده می‌شود. این پروتوکول عمدتاً در ساختارهای client-server مورد استفاده است و کاربران را قادر می‌سازد فایل‌ها را در یک سیستم از راه دور (Remote System) مشاهده، ذخیره و به‌روزرسانی کنند. برای استفاده از پروتوکول چندین پیش‌نیاز وجود دارد و در کار با سیستم‌های لینوکس راحت‌تر است، زیرا این پروتوکول در هستهٔ لینوکس گنجانیده شده است. این یک پروتوکول محبوب و عمومی دسترسی به فایل‌سیستم است که با Linux، FreeBSD، Apple's macOS، Solaris و AIX کار می‌کند. غیر از این، پروتوکول‌های دیگری هم مانند SMB یا Server Message

Block که CIFS هم شناخته می‌شود، پروتوکول AFP یا Apple Filing Protocol، پروتوکول NCP یا Network Core Protocol هم به فایل‌سیستم دسترسی دارند. NFS یک استندرد برای ذخیره‌سازی شبکه متصل به شبکهٔ NAS است. پروتوکول به کاربران اجازه می‌دهد فایل‌ها را در یک شبکه از راه دور مشاهده، ذخیره و به‌روز کنند. SAMBA مناسب کاربران ویندوزی و برای کاربران لینوکس و یونیکس، NFS گزینهٔ مناسبی است.

۵.۳ فایل‌سرور سمبا (Samba File Server)

درواقع پروتوکول SMB همان CIFS می‌باشد که یک پروتوکول برای دسترسی و به اشتراک‌گذاری فایل‌ها، پرنترها و پورت‌ها در شبکه می‌باشد و ارتباطات گوناگون بین دستگاه از طریق SMB پروتوکول صورت می‌گیرد. SMB همچنین یک مکانیزم IPC تصدیق هویت شده (Authenticated Inter-Process Communication) را فراهم می‌کند. پروتوکول SMB یا Server Message Block یک پروتوکول لایهٔ ۷ می‌باشد که در سیستم‌عامل قرار دارد. پروتوکول Server Message Block می‌تواند با لایهٔ Session (و لایه‌های پایین‌تر) به راه‌های گوناگونی فعالیت کند:

- مستقیماً روی پورت ۴۴۵ TCP؛
- از طریق API مربوط به NetBIOS، که در نتیجه می‌تواند روی چند پروتوکول لایه Transport نیز فعالیت کند.
- روی پورت‌های 137,138 UDP و 137,139 NetBIOS over TCP/IP؛

SMB(Server Message Block) یا سامبا در سال ۱۹۹۲ توسط Andrew Tridge آغاز به کار کرد. Samba پروتوکول اشتراک‌گذاری فایل‌ها و منابع موجود در شبکه با هدف دسترسی سریع و آسان به منابع به اشتراک گذاشته شده در شبکه می‌باشد (Negus, ۲۰۱۰).

راه‌اندازی یک سرور Samba برای کسانی که می‌خواهند فایل‌های خود را در یک جا متمرکز کرده و از تمام سیستم‌عامل‌ها (از جمله windows, linux , android ,..) به آن دسترسی داشته باشند، مفید به استفاده است.

برای راه‌اندازی یک فایل‌سرور بر اساس samba ابتدا بسته نرم‌افزاری samba را باید نصب کنید.

```
yum install samba
```

فایل تنظیمات etc/samba/smb.conf/ می‌باشد. فایل smb.conf دارای بخش‌های مختلفی است که در ادامه به تشریح آنها می‌پردازیم.

- [global]: در این بخش پیکربندی کلی و مورد استفاده در تمامی بخش‌ها تعریف می‌گردد.
- [share]: یک اشتراک را تعریف می‌کند.

- **[printers]**: در این بخش پرنترهای اشتراک گذاشته تعریف می‌شوند.
- **[homes]**: دایرکتری home کاربران اشتراک گذاشته تعریف می‌شوند.

در داخل هر یک از این بخش‌ها پارامترهایی وجود دارد که هر پارامتر می‌تواند قیمت‌های true, false, yes, no ویا یک رشته (string) یا مسیر (path) داشته باشد.

در ادامه به تشریح پارامترهای مهمتر می‌پردازیم:

- **browseable**: تعیین می‌کند که اشتراک مورد نظرمان می‌تواند Browse شود ویا خیر.
- **comment**: متن توضیح که در کنار اشتراک یا سرویس نمایش داده می‌شود.
- **create mask**: برای تبدیل مجوزهای ویندوز به یونیکس/ لینوکس در هنگام ایجاد فایل به‌کار می‌رود. مقدار پیش‌فرض برابر 0744 است.
- **directory mask**: برای تبدیل مجوزهای ویندوز به یونیکس/سطروکس در هنگام ایجاد دایرکتری به‌کار می‌رود. مقدار پیش‌فرض برابر 755 است.
- **encrypt passwords**: تعیین می‌کند که در هنگام Authentication بین سرویس‌دهنده و سرویس‌گیرنده کلمات عبور رمزنگاری شوند یا خیر.
- **security**: این گزینه مشخص می‌کند که کاربران چگونه به سرور متصل شده و تصدیق هویت شوند. مقدار پیش‌فرض برابر user است.
- **read only**: امکان ویرایش فایل‌ها توسط کاربران را مشخص می‌کند.
- **invalid users**: این کاربران تعریف شده قادر نخواهند بود به سیستم وارد شوند.
- **workgroup**: تعیین‌کنندهٔ نام شبکه Workgroup است.
- **server string**: نام سرور را در سیستم‌های سرویس‌گیرنده مشخص می‌کند.
- **interfaces**: آدرس‌هایی را که مجاز به برقراری ارتباط با سرور هستند، مشخص می‌کند.
- **printable**: امکان ارسال چاپ توسط کاربر به سرور را مشخص می‌کند.

یک فولدر با دستور زیر جهت اشتراک‌گذاری ایجاد می‌کنیم:

```
mkdir -p /home/ahmad
```

سطح دسترسی خواندن و نوشتن هم با دستور زیر برای فولدر ahmad ایجاد می‌کنیم:

```
Chmode –R 755 /home/ahmad/
```

با دستور زیر هم گروه کاربری فولدر ahmad را ویرایش می‌کنیم:

```
chown -R nobody:nogroup /home/ahmad/
```

سپس فایل smb.conf را با استفاده از یک ویرایشگر متن (Text editor) باز کرده:

```
sudo vim /etc/samba/smb.conf
```

در قسمت Global Settings کُدهای زیر را وارد نمایید:

```
[global]
workgroup = WORKGROUP
server string = Samba Server %v
netbios name = debian
security = user
map to guest = bad user
dns proxy = no


[ahmad]
path = /home/ahmad
browsable =yes
writable = yes
guest ok = no
read only = no
```

فایل را ذخیره نمایید سپس سرویس samba را با استفاده از دستور زیر restart کنید:

```
Service samba restart
```

۵.۳.۱ محدودیت دسترسی

تنظیماتی که در بالا اعمال شد تمام کاربران می‌توان بدون هیچ محدودیت و کنترولی اقدام به ایجاد فایل ویا حذف فایل‌های به اشتراک‌گذاشته‌شده نمایند. به همین منظور برای محدودکردن دسترسی کاربران می‌توانیم با گذاشتن پس‌ورد در فایل به اشتراک‌گذاشته‌شده دسترسی یک تعداد از کاربران را به فایل به اشتراک‌گذاشته‌شده محدود کنیم و فقط برای گروه خاصی که به این کاربر و پس‌ورد دسترسی دارند فایل‌ها قابل دسترس باشد.

ابتدا یک یوزر برای samba ایجاد می‌کنیم:

```
Sudo useradd mohammad
```

سپس با دستور زیر برای کاربر ساخته شده پس‌ورد بدهید:

```
Sudo smbpasswd –a mohammad
New SMB password
Retype new SMB password
```

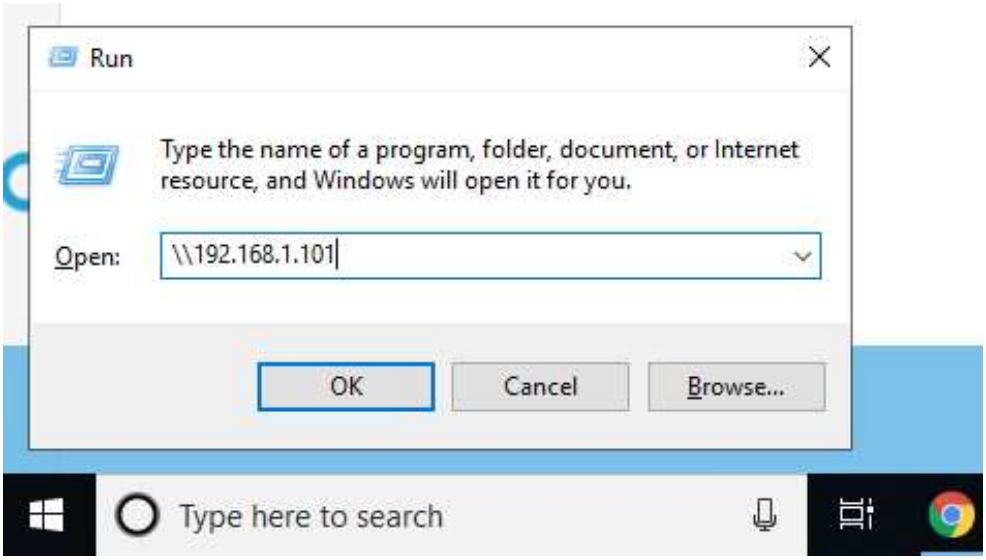
یکبار سرویس samba را با دستور زیر restart کنید:

```
Service samba restart
```

۵.۳.۲ دسترسی به File Shareing در Windows

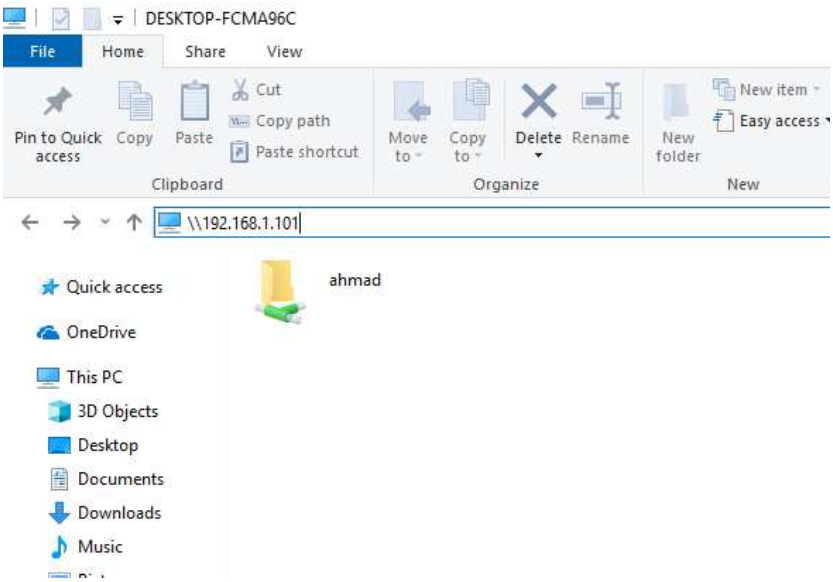
جهت دسترسی به فولدرهای به اشتراک‌گذاشته‌شده در ویندوز در مینوی RUN یا داخل آدرس بار MyComputer تان آدرس سرور را وارد می‌کنیم. به ترتیب زیر:

```
\\192.168.1.101
```



شکل(۵-۱) دسترسی به سرور samba از windows

اگر فایل به اشتراک‌گذاشته‌شده پس‌ورد نداشته باشد به راحتی می‌توانید بعد از واردکردن ip سرور وارد فولدر به اشتراک‌گذاشته شده شوید:



شکل(۵-۲) دسترسی به سرور samba از windows explorer

در غیر این صورت اگر پس‌ورد تعریف کرده باشید پیغامی مبنی بر این‌که کاربر و پس‌ورد را وارد کنید برای شما نشان داده خواهد شد که با واردکردن کاربر و پس‌وردی که در مرحلهٔ قبل ایجاد کردیم، می‌توانیم وارد فایل به اشتراک‌گذاشته‌شده شویم:

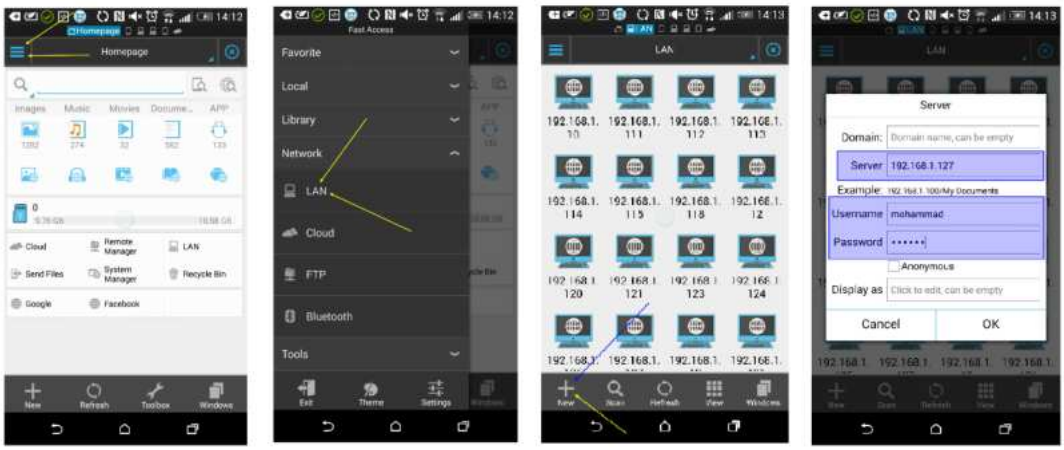
شکل(۵-۳) وارد به سرور samba

۵.۳.۳ دسترسی به File Shareing در Android

از آنجایی‌که هسته سیستم‌عامل Android از لینوکس گرفته شده است، به راحتی می‌تواند با سیستم‌عامل لینوکس ارتباط بر قرار کند. برای دسترسی به سرور Samba در Android ابتدا باید یک برنامهٔ مدیریت فایل را که قابلیت دسترسی به فایل‌سیستم شبکه را نیز داشته باشد، نصب کنید؛ برای نمونه می‌توان ES File Explorer استفاده کرد که از طریق play store قابل نصب است.

بعد از نصب این برنامه آن را باز کنید و از مینوی اصلی برنامه در بخش LAN رفته و یک کمپیوتر جدید را اضافه کنید. این کمپیوتر همان سرور Samba ما خواهد بود، چون ما می‌خواهیم به فایل‌های به

اشتراک‌گذاشته‌شدهٔ سرور samba ساخته‌شدهٔ خود دسترسی پیدا کنیم، در شکل زیر این پروسه نشان داده شده است:



شکل(۵-۴) طریقه اتصال به سرور samba از android

۵.۴ عیارسازی سرور NFS

پروتوکول NFS برای اولین بار در سال ۱۹۸۴ توسط شرکت Sun Microsystems ارائه شده و تا به‌حال تغییرات زیادی نموده است. این پروتوکول اساساً برای سیستم-عامل‌های خانواده یونیکس کاربرد داشته و گسترش یافته است ولی اکنون به‌عنوان یک استندرد برای سیستم‌های ناهمگون(heterogeneous) تبدیل شده است. با استفاده از این پروتوکول مشتریان می‌توانند با فایل‌های موجود در شبکه رفتاری مشابه با فایل‌های ذخیره شده در دسک‌های ذخیره‌سازی محلی داشته باشند. به عبارت دیگر این سرویس امکانی را فراهم می‌آورد که با استفاده از آن می‌توان به فایل‌های موجود در شبکه همانند فایل‌های ذخیره شده درهارددسک معمولی دسترسی داشت و از آنها استفاده نمود(Soyinka,۲۰۱۶).

در NFS عملیات دسترسی به فایل مشترک با رد و بدل نمودن یک سری پیغام در هر دو سوی سرویس‌دهنده و سرویس‌گیرنده صورت می‌گیرد. همان‌طورکه بیان شد، NFS از مدل Client/Server در تعریف سیستم‌ها استفاده می‌نماید و باعث تحولات اساسی در سیستم‌های مبتنی بر یونیکس شده است چرا که هر سیستم می‌تواند به‌عنوان یک سرویس‌دهنده امکان دسترسی به فایل‌های خود را به سیستم‌های دیگر بدهد.

این سیستم‌فایل شبکه مشابه map drive در سیستم‌عامل windows می‌ماند که در آن شما فایل‌سیستم یک کمپیوتر دیگر شبکه را به‌عنوان بخشی از فایل‌سیستم محلی (local) به کمپیوتر خود اضافه می‌کنید، یعنی از نظر فزیکی فایل‌های شما در یک کمپیوتر دیگر ذخیره می‌شود ولی از نظر منطقی شما آنها را در سیستم خود مشاهده می‌کنید و همانند فایل‌سیستم محلی خود با آنها رفتار می‌کنید و این قابلیت بسیار جالب توجهی است.

شما در مدیریت سیستم یک با نحوهٔ mountکردن یا unmountکردن بخشی از فایل‌سیستم آشنا شدید، mount یک دستور است که از طریق آن شما می‌توانید یک فایل‌سیستم خارجی را در هر جایی از فایل‌سیستم فعلی خود اضافه (یا اصطلاحا map) کنید؛ به‌عنوان مثال: اگر شما بخواهید CD-ROM خود را به‌صورت دستی به فایل‌سیستم اضافه کنید و بتوانید فایل‌های آن را مشاهده کنید باید دستور زیر را استفاده کنید:

```
$ mount -t iso9660 /dev/scd0 /media/cdrom
```

در این صورت شما در فایل‌سیستم خود در آدرس /media/cdrom می‌توانید محتویات cd را مشاهده کنید، به این پروسه mounting گفته می‌شود و به عکس این پروسه unmounting می‌شود. که با دستورumount /dev/scd0 یا eject /dev/scd0 می‌توانید این کار را انجام دهید. به همین شکل هم شما می‌توانید که فایل‌سیستم NFS را به سیستم خود اضافه کنید ولی تفاوت اساسی در این است که این فایل‌سیستم تحت شبکه است و از یک کمپیوتر دیگر به سیستم شما اضافه می‌شود.

با توجه به آنچه که تذکر داده شد، NFS به‌عنوان یک سیستم‌فایل توزیع‌شده برای به اشتراک‌گذاشتن فایل‌ها و دایرکتری‌ها بین سیستم‌عامل‌های مختلف ایجاد گردیده است. این سیستم به کاربر اجازه می‌دهد تا به فایل‌های روی شبکه همانند فایل‌های محلی دسترسی پیدا نمایند (درخواست mount را در سطح یک دایرکتری و تمام زیردایرکتری‌های مربوطه به سرویس‌دهنده می‌دهد). بنابراین امکان mount شدن یک فایل‌سیستم محلی روی یک شبکه و میزبان‌های دوردست وجود دارد (به‌طوری‌که گویا به‌صورت محلی در سیستم یکسان mount شده‌اند). بنابراین به کمک این سیستم، اشتراک فایل بین سیستم‌عامل‌های مختلف یونیکس به لینوکس و برعکس به راحتی امکان‌پذیر می‌باشد.

۵.۴.۱ نحوهٔ کار NFS

برای درک بهتر اجازه دهید که یک مثال بزنیم. فرض کنید یک سرور NFS به نام ServerA و یک کلاینت NFS به نام ClientA داریم. ServerA ضروت به اشتراک‌گذاری /home در شبکه را دارد که به این کار اصطلاحاً صادرکردن (exporting) پارتیشن /home گفته می‌شود. و ClientA می‌خواهد به محتویات و فایل‌های پارتیشن صادرشده دسترسی پیدا کند. فرض کنید تمام تنظیمات انجام شده است. ClientA برای دسترسی به این پارتیشن به اشتراک‌گذاشته‌شده، باید دستوری مشابه دستور زیر را در ترمینال وارد نمایید:

```
[root@clientA ~]# mount serverA:/home /home
```

البته دستور واقعی دقیقاً این نیست، تمام کاربرانی که در ClientA هستند قادر خواهند بود تا محتویات /home را مشاهده کنند. البته توجه داشته باشید که این پارتیشن در ServerA موقعیت دارد ولی در فایل‌سیستم ClientA از آدرس /home ویا هر جای دیگری که mount شده باشد در دسترس خواهد بود.

NFS از طریق مکانیزمی به نام RPC[1] عمل می‌کند. هر عملیه‌یی که کاربر در ClientA بر روی فایل‌های اشتراک‌گذاری‌شده انجام می‌دهد از طریق RPC به سرور انتقال می‌یابد تا سرور آنها را واقعاً عملی سازد. هر سرویسی که در سیستم‌عامل لینوکس نصب می‌شود باید خود را در مدیریت سرویس RPC به نام portmap راجستر کند. سرویس portmap مسئولیت دارد تا به کلاینت‌ها بگوید که سرویس‌ها در کدام پورت از سرور فعال هستند.

۵.۴.۲ نسخه‌های مختلف NFS

در زمان نوشتن این کتاب سه نسخهٔ شناخته شده از این پروتوکول وجود دارد، NFSv2, NFSv3, NFSv4 که جدیدترین آنها NFSv4 می‌باشد.

NFSv2: از TCP یا UDP برای انتقال استفاده می‌کند، کاربران فقط می‌توانستند که به فایل‌های کمتر از 2GB دسترسی داشته باشند. همچنان درخواست mount بر اساس host تضمین می‌شود نه بر اساس کاربر.

NFSv3: از TCP یا UDP برای انتقال استفاده می‌کند، بر خلاف نسخهٔ قبلی هیچ محدودیتی برای کاربران از نظر اندازهٔ فایل‌ها جهت انتقال و دسترسی وجود ندارد. همچنان درخواست mount بر per-user تضمین شده است.

NFSv4: از TCP و SCTP[2] برای انتقال استفاده می‌کند، از قابلیت‌های امنیتی بالاتری بر خوردار است و از Kerberos (نوع از Autentaction protocol است که برای ارتباط کمپیوتر در شبکه استفاده می‌شود) نیز پشتیبانی می‌کند. از پورت شناخته‌شدهٔ 2049 استفاده می‌کند. در این نسخه دیگر از پروتوکول NFS جداگانه (rpc.mountd, rpc.lockd, rpc.statd) استفاده نمی‌کند زیرا این قابلیت‌ها این نسخه تمام این قابلیت‌ها را تحت یک پروتوکول یکجا ساخته است. سرویس portmap (مدیریت‌کنندهٔ درخواست‌ها RPC) دیگر استفاده نمی‌شود. از لست کنترول دسترسی (ACL[3]) پشتیبانی می‌کند. این نسخه مفهوم شبه فایل‌سیستم (psedu-file system ـ نوع فایل‌سیستم را که دارای محتویات حقیقی نبوده مجازی می‌باشد مانند proc/) معرفی کرد.

نسخه‌های NFS در هنگام استفاده از دستور mount قابل انتخاب است برای mountکردن NFSv2 از گزینهٔ nfsvers=2 استفاده می‌شود. NFSv3 از گزینهٔ nfsvers=3 استفاده می‌شود ولی برای NFSv4 از گزینهٔ nfsvers دیگر استفاده نمی‌شود، فقط از nfs4 استفاده می‌شود.

---

[1] - Remote Procedure Call

[2] - Stream Control Transmission Protocol

[3] - Access Control List

۵.۴.۳ مزایای NFS

- NFS دسترسی local به فایل‌های remote را امکان پذیر می‌سازد.
- با استفاده از NFS نیاز نیست که سیستم‌عامل‌ها یکسان باشند.
- با کمک NFS ما می‌توانیم راهکارهای centralized storage را انجام دهیم.
- کاربران به معلومات و دیتاهای خود بدون توجه به موقعیت فزیکی آنها دسترسی دارند.
- تغییرات و در فایل‌سیستم برای تمام کابران به‌نگام (synchronize) می‌شود.
- نسخه‌های جدیدتری از NFS همچنین pseudo root mounts, acl پشتیبانی می کند.
- می‌توان آن را با استفاده از Firewalls و Kerberos امن ساخت.

۵.۴.۴ سرویس‌های NFS

بستۀ نرم‌افزاری سرور NFS شامل سه بسته می‌باشد:

- portmap: این بسته ارتباطات ایجادشده از ماشین‌های دیگر را به سرویس RPC درست map می کند. (بدون نیاز به NFS v4)
- NFS: این بسته درخواست‌های file sharing ریموت (remote) را به درخواست‌های local ترجمه (Translate) می کند.
- rpc.mountd: این سرویس مسئول mount و unmountکردن file system می‌باشد.

۵.۴.۵ فایل‌های مهم برای تنظیمات NFS

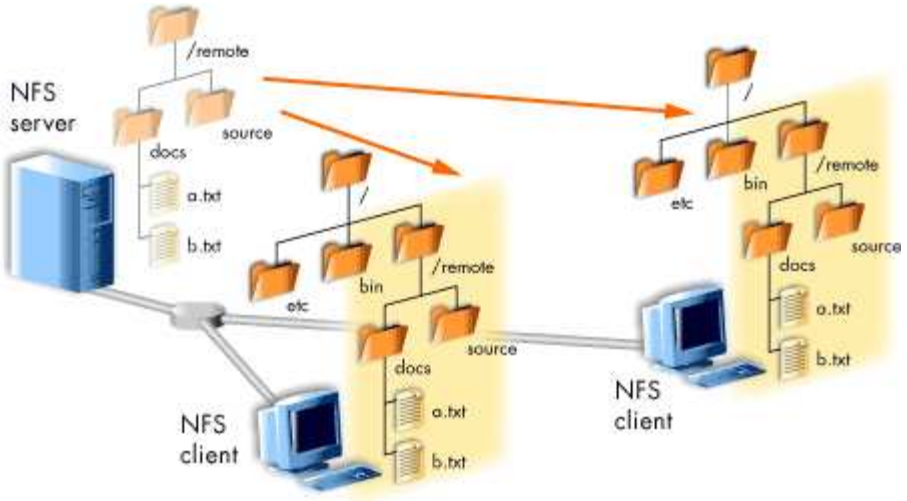
- etc/export: این فایل تنظیمات اصلی NFS می‌باشد و تمام فایل‌های export شده و دایرکتری‌ها در این فایل و در سمت سرور NFS تعریف می‌شود.
- etc/fstab: برای mountکردن یک دایرکتری NFS روی سیستم حتی بعد از reboot ما نیاز داریم که در فایل etc/fstab تنظیماتی را صورت دهیم.
- etc/sysconfig/nfs : تنظیمات NFS برای کنترول‌کردن این که rpc و سرویس‌های دیگر روی چه پورتی lisening می کنند.

۵.۵ عیارسازی سرور NFS در لینوکس

برای راه‌اندازی NFS حد اقل به ۲ سیستم Linux/Unix نیاز داریم. دو کمپیوتر را در در نظر می‌گیریم.

NFS Server: nfsserver.example.com و IP-192.168.0.100

NFS Client: nfsclient.example.com و IP-192.168.0.101



شکل(۵-۵) سرور NFS مسیر /remote را برای دو کلاینت به اشتراک گذاشته است

هدف ما راه‌اندازی سناریویی است که در تصویر فوق مشاهده می‌کنید.

۵.۵.۱ نصب NFS Server و NFS Client

در ابتدا باید بسته‌های نرم‌افزاری NFS هم در سرور NFS و هم در کلاینت NFS نصب شوند. می‌توانیم با استفاده از YUM (در سیستم‌عامل‌های خانوادهٔ Red Hat) و apt-get (در سیستم‌عامل‌های خانوادهٔ Debian) آن را نصب کنیم:

```
[root@nfsserver ~]# yum install nfs-utils nfs-utils-lib
[root@nfsserver ~]# yum install portmap (not required with NFSv4)
```

دستور نصب برای سرور Ubuntu به شکل زیر است:

```
[root@nfsserver ~]# apt-get install nfs-utils nfs-utils-lib
```

اکنون روی هر ۲ سیستم سرویس‌ها را Start می‌کنیم:

```
[root@nfsserver ~]# /etc/init.d/portmap start
[root@nfsserver ~]# /etc/init.d/nfs start
[root@nfsserver ~]# chkconfig --level 35 portmap on
[root@nfsserver ~]# chkconfig --level 35 nfs on
```

بعد از این باید آنها را برای اشتراک‌گذاری فایل‌ها تنظیم نماییم.

۵.۵.۲ تنظیمات سرور NFS

برای به اشتراک‌گذاشتن یک دایرکتری با استفاده از NFS باید در فایل etc/exports تغییراتی را ایجاد کنیم. در اینجا دایرکتری‌یی با نام nfsshare در پارتیشن / (اسلش) ایجاد می‌کنم و از آن برای به اشتراک‌گذاشتن فایل‌های با کلاینت استفاده خواهم کرد. شما می‌توانید حتی یک دایرکتری موجود روی سیستم‌تان را با NFS به اشتراک بگذارید:

```
[root@nfsserver ~]# mkdir /nfsshare
```

اکنون نیاز دارید تغییری را در فایل etc/exports داشته باشیم و سپس سرویس را به منظور اعمال تغییرات باید reset کنیم:

```
[root@nfsserver ~]# vi /etc/exports
/nfsshare 192.168.0.101(rw,sync,no_root_squash)
```

در مثال بالا دایرکتری nfsshare در پارتیشن / (اسلش) با کلاینت ۱۹۲.۱۶۸.۰.۱۰۱ و با سطح دسترسی read/write یا (rw) به اشتراک گذاشته شده است، شما حتی می‌توانید به جای نوشتن ip در مثال بالا hostname ویا دومین را استفاده کنید.

۵.۵.۳ گزینه‌های NFS

بعضی از گزینه‌هایی که می‌توانیم در فایل etc/exports استفاده کنیم، قرار زیر می‌باشد:

ro: با کمک این گزینه می‌توانیم دسترسی read only را برای فایل‌های به اشتراک‌گذاشته‌شده فعال بسازیم تا کلاینت فقط بتواند فایل‌ها را بخواند و تغییرات آورده نمی تواند.

rw: این گزینه به client server اجازه می‌دهد تا علاوه بر دسترسی read بتواند دسترسی write هم داشته باشد.

sync: دایرکتری به اشتراک‌گذاشته‌شده را به محض این‌که تغییرات لحاظ شوند، تأیید و به‌هنگام (synnchronize) می‌کند.

no_subtree_check: این گزینه از چک subtree جلوگیری می‌کند. زمانی‌که یک دایرکتری به اشتراک گذاشته‌شده در حقیقت زیردایرکتری‌یی از یک فایل‌سیستم بزرگتر باشد NFS از هر دایرکتری بالای آن scan (اسکن) را انجام خواهد داد. این اسکن به منظور بررسی سطح دسترسی (permission) و جزئیات می‌باشد.

غیر فعال ساختن چک subtree اگرچه ممکن است reliability را در NFS افزایش دهد اما امنیت (security) را کاهش خواهد داد.

no_root_squash: به این معنا است که کاربر root در کلاینت تمام صلاحیت‌های کاربر root واقعی را بر روی فایل‌های اشتراک‌گذاری‌شده داشته باشد.

root_squash: این گزینه عکس گزینهٔ قبلی است.

برای معلومات در مورد تمام گزینه‌ها می‌توانید راهنمای export را مطالعه کنید (Negus,۲۰۱۰).

```
$man export
```

۵.۵.۴ تنظیمات NFS Client

بعد از تنظیم NFS Server باید دایرکتری یا پارتیشن به اشتراک‌گذاشته‌شده را در کلاینت mount کنیم. ابتدا چک می‌کنیم که در سرور ۱۹۲.۱۶۸.۰.۱۰۰ چه چیزهایی به اشتراک گذاشته شده است:

```
[root@nfsclient ~]# showmount -e 192.168.0.100
Export list for 192.168.0.100:
/nfsshare 192.168.0.101
```

دستور بالا نشان می‌دهد که یک دایرکتری با نام nfsshare در ۱۹۲.۱۶۸.۰.۱۰۰ به اشتراک گذاشته شده است.

برای mountکردن دایرکتری مورد نظر می‌توانیم دستور زیر را استفاده کنیم:

```
[root@nfsclient ~]# mount -t nfs 192.168.0.100:/nfsshare /mnt/nfsshare
```

دستور بالا دایرکتری مورد نظر را در مسیر mnt/nfsshare در کلاینت mount می‌کند. شما می‌توانید با استفاده از دستور زیر آن را بررسی کنید:

```
[root@nfsclient ~]# mount | grep nfs
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
nfsd on /proc/fs/nfsd type nfsd (rw)
192.168.0.100:/nfsshare on /mnt type nfs (rw,addr=192.168.0.100)
```

اما با خاموش شدن سیستم کلاینت این اشتراک‌گذاری از بین می‌رود و باید مجدداً باید آن را mount کنید. برای mountکردن دایمی دایرکتری به اشتراک‌گذاشته‌شده حتی بعد از reboot شما باید خطی را در فایل etc/fstab ایجاد کنید:

```
[root@nfsclient ~]# vi /etc/fstab
```

و خط جدید زیر را همان‌طور که می‌بینید به انتهای فایل اضافه کنید:

```
192.168.0.100:/nfsshare /mnt nfs defaults 0 0
```

۵.۵.۵ تست تنظیمات NFS

برای امتحان‌کردن تنظیمات و اطمینان از صحت کار سرور NFS یک فایل در مسیر اشتراک /nfsshare در سرور ایجاد می‌کنیم و از طرف کلاینت آن را باز می‌کنیم.

در فایل‌سرور متنی ساده به نام nfstest.txt در مسیر اشتراک‌گذاری‌شده /nfsshare ایجاد می‌کنیم:

```
[root@nfsserver ~]# cat > /nfsshare/nfstest.txt
This is a test file to test the working of NFS server setup.
```

در کلاینت به آدرس /mnt/nfsshare رفته تا وجود فایل جدید را چک کنیم:

```
[root@nfsclient]# ll /mnt/nfsshare
total 4
-rw-r--r-- 1 root root 61 Sep 21 21:44 nfstest.txt
root@nfsclient ~]# cat /mnt/nfsshare/nfstest.txt
This is a test file to test the working of NFS server setup.
```

۵.۵.۶ حذف NFS Mount

اگر شما بعد از اتمام file sharing بخواهید دایرکتری به اشتراک‌گذاشته‌شده را از سرور unmount کنید، می‌توانید به آسانی از دستور umount استفاده کنید:

```
root@nfsclient ~]# umount /mnt/nfsshare
```

برای اطمینان از حذف کُد زیر را اجرا نمایید:

```
[root@nfsclient ~]# df -h -F nfs
```

همان‌طور که میبینید دایرکتری اشتراک‌گذاری شده دیگر موجود نیستند.

۵.۵.۷ دستورات مهم NFS

بعضی از دستورات مهم برای NFS به شرح زیر است:

showmount –e: تمام فایل‌های اشتراک گذاشته شده کمپیوتر شما را نمایش می‌دهد.

showmount –e <server–ip or hostname>: فایل‌های اشتراک‌گذاری شده در یک سرور دیگر را نمایش می‌دهد.

showmount –d : لست تمام subdirectoryهای مسیر اشتراک‌گذاری‌شده را نمایش می‌دهد.

exportfs –v : لستی از فایل‌های اشتراک‌گذاری‌شده در یک سرور و گزینه‌های موجود را نمایش می‌دهد.

exportfs –a: تمامی دایرکتری‌های اشتراک‌گذاری‌شده که در فایل etc/exports تعیین شده است را به اشترک می‌گذارد.

exportfs –u: تمام دایرکتری‌هایی که در فایل etc/exports/ تعیین شده و به اشتراک گذاشته شده‌اند را از حالت اشتراک خارج می‌سازد.

exportfs –r : بعد از تغییر فایل etc/exports سرور را refresh می‌کند.

خلاصهٔ فصل پنجم

- برای راه‌اندازی یک فایل‌سرور جنبه‌های مختلفی باید لحاظ شود، تکنالوژی‌های ذخیره‌سازی مثل NAS, SAN، دیزاین شبکه و پروتوکول‌های اشتراک‌گذاری فایل از مهمترین این جنبه‌ها است.
- از مهمترین پروتوکول‌های اشتراک‌گذاری فایل FTP, NFS و Samba است.
- NFS یک استندرد برای ذخیره‌سازی شبکه متصل به شبکه NAS است. پروتوکول به کاربران اجازه می‌دهد فایل‌ها را در یک شبکه از راه دور مشاهده، ذخیره و به‌روز کنند. Samba مناسب کاربران ویندوزی و برای کاربران لینوکس و یونیکس، NFS گزینهٔ مناسبی است.
- پروتوکول Samba به‌طور پیش‌فرض روی پورت TCP ۴۴۵ فعالیت می‌کند، بر اساس پروتوکول NetBIOS عمل می‌کند. همچنان روی پورت‌های UDP 137,138 و TCP 137,139 NetBIOS over TCP/IP نیز عمل می‌کند. فایل تنظیمات samba در /etc/samba/smb.conf می‌باشد.
- سه نسخهٔ شناخته‌شده از این پروتوکول وجود دارد، NFSv2, NFSv3, NFSv4 که جدیدترین آنها NFSv4 می‌باشد. NFS از طریق مکانیزمی به نام RPC[1] عمل می‌کند.
- NFSv4 که آخرین نسخه‌های NFS است، از TCP و SCTP[2] برای انتقال استفاده می‌کند، از قابلیت‌های امنیتی بالاتری بر خوردار است و از Kerberos نیز پشتیبانی می‌کند. از پورت شناخته شده 2049 استفاده می‌کند. در این نسخه دیگر از پروتوکول NFS جداگانه (rpc.mountd, rpc.lockd, rpc.statd) استفاده نمی‌کند زیرا این نسخه تمام این قابلیت‌ها را تحت یک پروتوکول یکجا ساخته است.

[1] - Remote Procedure Call

[2] - Stream Control Transmission Protocol

سوالات و فعالیت های فصل پنجم

۱. فایل‌سرور را تعریف کنید.

۲. پروتوکول‌های معروف مورد استفاده در فایل‌سرور را نام برده و تشریح نمایید.

۳. پروسه عیارسازی سرور Samba را تشریح کنید.

۴. گزینه‌های NFS را نام برده و تشریح کنید.

۵. نسخه‌های مختلف NFS را با هم مقایسه کنید.

۶. فایل‌های مهم برای تنظیمات NFS را نام ببرید.

فعالیت ها

۱. با جستجو در انترنت در مورد SAN و NAS که از جمله تکنالوژی‌های ذخیره‌سازی می‌باشند، معلومات جمع‌آوری نمایید و آن را در صنف به بحث بگذارید.

۲. با جستجو در انترنت دو پروتوکول اشتراک فایل (غیر از آنچه در این فصل بیان شد) را یافته و در مورد آنها معلومات جمع‌آوری نمایید.

۳. پروسهٔ نصب و عیارسازی Samba را بر اساس مطالبی که از این فصل آموخته‌اید، به شکل عملی تمرین نمایید و مشکلاتی را که با آن مواجه شده‌اید، نوت گرفته و در صنف به بحث بگذارید.

۴. از طریق یک سیستم‌عامل windows تلاش کنید که به سرور sambaیی که در فعالیت قبل ساخته‌اید دسترسی پیدا کنید (نحوهٔ دسترسی بالعکس را نیز تمرین نمایید).

۵. بر روی سروری که در فعالیت شماره ۳، سرویس Samba را فعال نمودید، سرویس NFS را نیز عیار بسازید. بعد از عیارسازی سرور دایرکتری home/ را به اشتراک بگذارید، از طریق یک کلاینت لینوکس آن را در آدرس mnt/homeshare/ اصطلاحاً mount کنید. از صحت کار سرور اطمینان حاصل نمایید.

۶. آیا بر روی یک سرور همزمان می‌توان چندین سرویس اشتراک فایل را عیار ساخت؟

فصل ششم

تنظیمات سایر سرویس‌های شبکه
(DHCP,FTP, Web Server)

هدف کلی: آشنایی و مهارت تنظیم سرویس‌های DHCP, FTP , Web در سیستم‌عامل سرور لینوکس.

اهداف آموزشی: در پایان این فصل محصلان قادر خواهند بود تا:

۱. سرویس DHCP و نحوهٔ عیارسازی آن را تشریح نمایند.
۲. مهارت عیارسازی و اشتراک فایل‌ها از طریق پروتوکول FTP کسب کنند.
۳. یک سیستم‌عامل سرور را به یک سرور Web تبدیل نمایند.

۶.۱ عیارسازی سرور DHCP

DHCP یک پروتوکول Server/client است که به‌صورت خودکار Clientها IP و معلوماتی نظیر Subnet و Default Getaway و DNS را ارائه می‌دهد.

RFC2132 و RFC2131 از DHCP به‌عنوان نیروی ضربتی مهندسی انترنت بر اساس پروتوکول Bootstrap تعریف می‌کند. پروتوکولی که با استفاده از DHCP بسیار از جزئیات را پیاده‌سازی می‌کند. DHCP به Client اجازه می‌دهد که تنظیمات مربوط به TCP/IP را از روی DHCP دریافت کنند.

DHCP یکی از سرویس‌هایی است که می‌توان در سیستم‌عامل سرور centos نصب و راه‌اندازی کرد و می‌تواند به تمام Client های دارای سیستم‌عامل اولیه، یک آدرس IP اختصاص بدهد. همچنین DHCP می‌تواند به‌عنوان یک سرویس در مودم‌های ADSL ، روترهای شبکه ویا سویچ‌های لایهٔ 3 اجرا شود (Negus, 2010).

DHCP یک سرویس برای کاهش پیچیدگی مدیریت آدرس‌دهی و تنظیمات است. تمام کمپیوترها و تجهیزات شبکه مبتنی بر TCP/IP باید یک آدرس IP داشته باشند. IP Addressها می‌توانند هم به‌صورت دستی (static) بر روی PCها تنظیم شوند و هم به‌صورت خودکار از یک DHCP Server آدرس IP دریافت کنند. بیشتر سیستم‌عامل‌ها به‌صورت پیش‌فرض به دنبال IP آدرس در شبکه هستند بنابراین هیچ تنظیماتی لازم نیست بر روی کلاینت‌ها صورت بگیرد، برای توضیع IP در یک شبکه با قابلیت DHCP قدم اول راه‌اندازی یک DHCP سرور است.

۶.۱.۱ پروتوکول DHCP

پروتوکول DHCP یکی دیگر از پروتوکول‌های مُدل TCP/IP می‌باشد که در لایهٔ application مورد استفاده قرار می‌گیرد و توسط سازمان IETF تحت RFC 2131 و RFC 3396 معرفی شده است. DHCP که مخفف کلمات Dynamic Host Configuration Protocol می‌باشد، یک پروتوکول ارتباطاتی است که مدیران شبکه را قادر به مدیریت مرکزی و خودکار IP addressها در شبکه می‌سازد. در یک شبکهٔ مبتنی بر IP، هر وسیله (device) که بخواهد به انترنت متصل شود، نیاز به یک IP address منحصربه‌فرد دارد. DHCP اجازه می‌دهد که مدیران یک شبکه نظارت مستقیمی بر توضیع IP addressها از یک نقطهٔ مرکزی داشته باشند و به‌طور خودکار هنگامی که کمپیوترهایی از محل‌های مختلف به شبکه متصل می‌شوند، بتوانند IP addressهای جدید را به آنها ارسال نمایند.

DHCP از مفهومی به نام lease استفاده می‌کند که به معنای اجاره‌کردن می‌باشد و به عبارتی دیگر مقدار زمانی است که یک IP address اختصاص‌یافته به یک کمپیوتر معتبر خواهد بود. زمان این اجارهٔ IP متفاوت خواهد بود و به‌طور مثال می‌تواند بسته به مدت زمانی باشد که یک کاربر احتیاج به اتصال انترنت در یک مکان خاص دارد که این مورد بیشتر در محیط‌های آموزشی مورد استفاده می‌باشد ویا سایر محیط‌هایی که

اغلب کاربران آن در حال تغییراند. این قابلیت یعنی استفاده از مدت زمان کوتاه اجارهٔ یک IP address، سرویس DHCP را قادر میسازد تا به‌طور خودکار عیارسازی مجدد را برای شبکه‌هایی که دارای تعداد زیادی کمپیوتر می‌باشند و به‌صورت متغیر در یک روز کاری مورد استفاده قرار می‌گیرند را برای IP addressها انجام دهد. البته ناگفته نماند که DHCP امکان پشتیبانی از آدرس‌های static (آدرس‌هایی که به‌صورت دستی تنظیم می‌شوند) را نیز دارد که این مورد برای کمپیوترهایی کاربرد دارد که حاوی Web server می‌باشند و نیاز به یک IP address دائم دارند.

DHCP درواقع جایگزینی برای دیگر پروتوکول‌های مدیریت IP addressها در شبکه‌های کمپیوتری بود؛ مانند: پروتوکول خود راه انداز dhcp. bootp یک پروتوکول پیشرفته‌تری می‌باشد به طوری که امکان جمع‌آوری IP addressها را بعد از ارائه‌دادن به کمپیوترها در اختیار دارد و از بروز هرگونه اختلال جلوگیری می‌کند. اما به هر حال هر دو پروتوکول یعنی DHCP, BOOTP را برای مدیریت و عیارسازی IP addressها می‌توان استفاده کرد. در ادامهٔ مطلب این‌که پیغام‌های DHCP در دیتاگرام‌های UDP حمل می‌شوند و در سمت server از شماره پورت 67 و در سمت client از پورت 68 استفاده می‌کند. پروتوکول‌هایی که در ارتباط با DHCP کار می‌کنند؛ شامل IP, BOOTP, UDP, TCP, RARP می‌باشند (Negus, 2010). در شکل زیر ساختار پروتوکول DHCP را مشاهده می‌نمایید:

8	16	24	32bit
Op	Htype	Hlen	Hops
Xid			
Secs		Flags	
Ciaddr			
Yiaddr			
Siaddr			
Giaddr			
Chaddr (16 bytes)			
Sname (64 bytes)			
File (128 bytes)			
Option (variable)			

شکل(۶-۱) هیدر پروتوکول DHCP

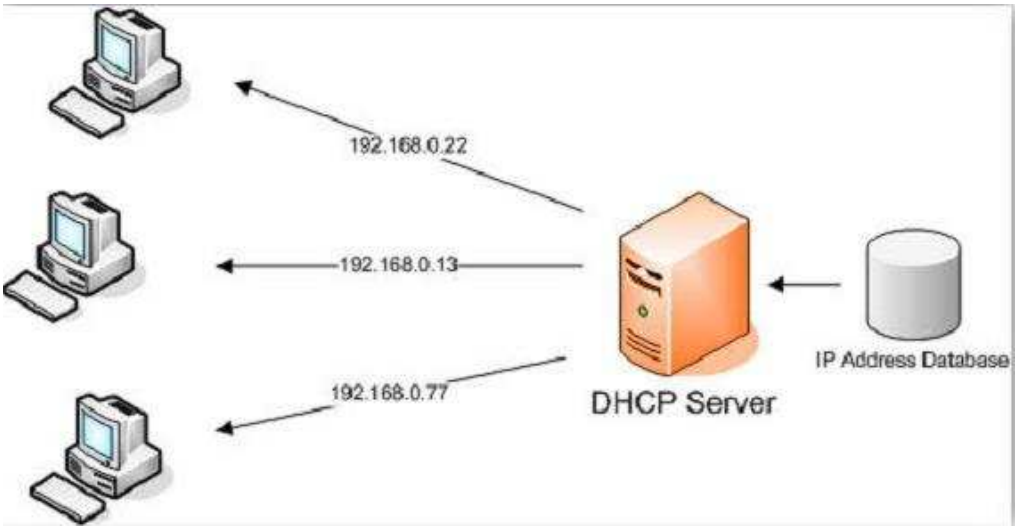
- **OP** کُد عملیاتی اختصاص‌یافته به پیام که می‌تواند BOOTREQUEST ویا BOOTREPLY باشد به عبارتی دیگر مشخص می‌کند که پیام از server تولید شده است یا client و اندازهٔ این پیام همان‌طورکه در جدول هم مشاهده می‌شود، bit8 که معادل یک بایت است.

- **Htype** نوع آدرس سخت‌افزاری موجود در بخش chaddr را مشخص می‌کند و اندازهٔ آن هم یک بایت است.
- **Hlen** طول آدرس سخت‌افزاری موجود در بخش Chaddr را بر حسب بایت نشان می‌دهد.
- **Hops** تعداد مسیریاب (Router)های موجود بین سرور و سرویس گیرنده را مشخص می‌کند و اندازهٔ آن یک بایت است.
- **Xid** یا transaction ID که حاوی یک شناسه برای نسبت‌دادن جواب‌ها به درخواست‌ها می‌باشد و به نوعی کُد متعلق به فرایند اختصاص‌یافته بین server و client می‌باشد و چهار بایت است.
- **Secs** مدتی را مشخص می‌سازد که یک کلاینت می‌خواهد IP خود را دوباره جدید ویا مدت زمان آن را تمدید نماید. و 2 بایت حجم آن است.
- **Flags DHCP Flags**: به یک هدف متفاوت استفاده می‌شود این Flags از طرف کلاینت تعین می‌شود و سایز آن۲ mb می‌باشد به DHCP server مشخص می‌کند که چگونه جواب کلاینت را ارسال نماید. که از سرور خواسته می‌شود که به عوض استفاده از پیام Unicost از پیام Broadcost استفاده نماید.

  Ciaddr آدرس IP کلاینت به عبارت دیگر آدرس IP کمپیوتر زمانی را که در وضعیت باند، تمدید اجارهٔ IP ویا ارتباط مجدد می‌باشد دارا است و اندازهٔ آن چهار بایت است.
- **Yiaddr**: آدرس IP کلاینت شما به عبارت دیگر آدرس IP را که توسط DHCP به یک کمپیوتر واگذار شده است، در بر دارد و اندازهٔ آن چهار بایت است.
- **Siaddr**: آدرس IP سرور بعدی را در یک دنبالهٔ Bootstrap مشخص می‌کند از این مقدار فقط زمانی‌که سرور DHCP یک فایل راه‌انداز اجرایی به یک client بدون دسک می‌دهد، استفاده می‌شود و اندازهٔ آن 4 بایت است.
- **DHCP Giaddr**: با مراجعه به این بخش که حاوی آدرس IP می‌باشد در جواب پیام DHCP Discover یک آدرس IP برای کلاینت‌ها انتخاب می‌کند.
- یک وا سط رلی‌کننده DHCP مستقر روی شبکه‌یی دیگر می‌باشد و اندازهٔ آن 4 بایت است.
- **Chaddr**: آدرس سخت‌افزاری client یا به عبارتی دیگر، با استفاده از نوع و اندازه‌یی که در بخش‌های htype و hlen مشخص شده است، نشان‌دهندهٔ آدرس سخت‌افزاری سرویس‌گیرنده می‌باشد. و مقدار آن 16 بایت است.
- **Sname** که حاوی نام DHCP server است. و مقدار آن 64 بایت است.
- **File**: شامل نام فایل boot، و به عبارتی برای clientهای بدون دسک حاوی نام و آدرس یک فایل راه‌انداز اجرایی می‌باشد و 128 بایت است.

- **Option**: فیلد پارامترهای اختیاری و به نوعی حاوی مجموعه‌یی از گزینه‌های DHCP می‌باشد. تا این‌که بهتر از DHCP استفاده گردد.

قبل از این‌که DHCP سرور برای کلاینت‌ها IP فراهم کند باید یک رنج IP برای سرور تعریف شود. این رنج به‌عنوان scope شناخته می‌شود که برای هر subnet فزیکی که در شبکهٔ شما است، یک تک IP پیشنهاد می‌دهد؛ به‌عنوان مثال: اگر شما دو subnet دارید DHCP Server شما باید به هر دوی آنها متصل باشد و شما برای هر subnet یک scope مجزا تعریف کنید.



شکل(۶-۲) جایگاه سرور DHCP در شبکه

۶.۱.۲ دلایل استفاده از DHCP

هر وسیله که در شبکه است باید یک IP منحصربه‌فرد برای دسترسی به شبکه و منابع آن داشته باشد بدون DHCP، آدرس IP برای کمپیوترهای جدید ویا pcهایی که از یک Subnet به Subnet دیگر انتقال یافته اند باید به‌صورت دستی تنظیم شود ویا آدرس IP برای PCهایی که از شبکهٔ ما حذف شده‌اند هم باید به‌صورت دستی اصلاح شود.

با سرویس DHCP تمام این فرآیند به‌صورت خودکار و مدیریت مرکزی اداره می‌شود. سرور DHCP آدرس‌های IP را در Pool خود ذخیره می‌کند و به هر کلاینت که در شبکه شروع به کار می‌کند، IP اجاره می‌دهد به‌خاطر این که آدرس‌های IP داینامیک اجاره‌یی نسبت به IP دستی مدت زمان طولانی استفاده نمی‌شود و به‌صورت خودکار برای دوباره اختصاص‌داده‌شدن به کلاینت‌ها به Pool برگردانده می‌شود.

مدیر شبکهٔ DHCP سروری را ایجاد می‌کند که شامل تنظیمات و معلومات مربوط به TCP/IP و همچنین تنظیمات آدرس‌دهی برای Clientها و پیشنهاد IPهای اجازه‌یی به آنها می‌شود.

سرور DHCP تنظیمات و معلومات را در یک دیتابیس که شامل موارد زیر است، ذخیره می‌کند:

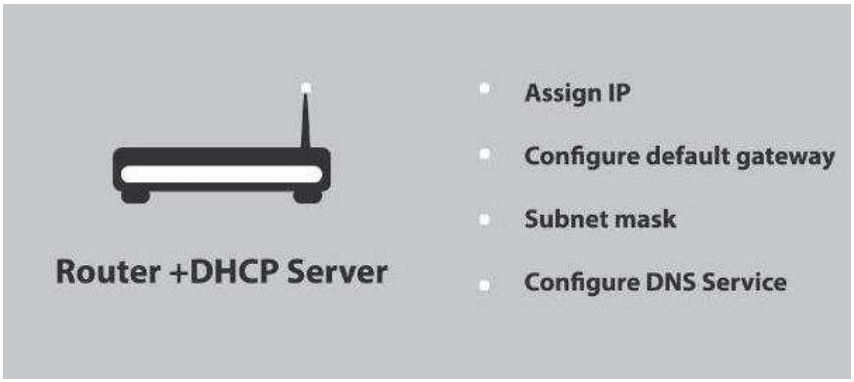
آدرس‌های IP معتبر که در pool حفظ می‌شود. (اختصاص‌دادن IP به کلاینت‌ها و همچنین آدرس‌های محدودشده برای اختصاص‌دادن به PCهای خاص).

مدت زمان اجاره یا طول مدتی که IP قرار است استفاده شود یا تمدید اجاره نامه.

آدرس DNS و Default Gatewayیی که همراه IP باید به کلاینت‌ها تخصیص داده شود.

۶.۱.۳ امکانات و خدمات DHCP

زمانی‌که شما یک DHCP سرور را در شبکهٔ خود راه‌اندازی می‌کنید، می‌توانید به‌صورت خودکار برای کلاینت‌ها و دیگر دستگاه‌های مبتنی بر TCP/IP آدرس IP تهیه کنید. این سرویس علاوه بر اختصاص‌دهندهٔ IP آدرس است، مدیریت تنظیمات شبکه برای default Gateway،subnet mask و سرویس DNS نیز بر عهده دارد. این سرویس هم در تجهیزاتی مثل سویچ و روتر قابل عیارسازی است و هم در سیستم‌عامل‌های سرور.



شکل(۶-۳) بخشی از خدمات DHCP

که به آنها اجازه می‌دهد که به منابع شبکه‌های دیگر دسترسی پیدا کنیم؛ مثل: DNS Server و Wins server این تکنولوژی Server/Client که سرور DHCP را قادر می‌سازد تا به کمپیوترها و دستگاه‌های دیگر که به‌عنوان DHCP Client هستند، IP آدرس اختصاص یا اجاره می‌دهد.

شما توسط سرویس DHCP می‌توانید موارد زیر را انجام دهید:

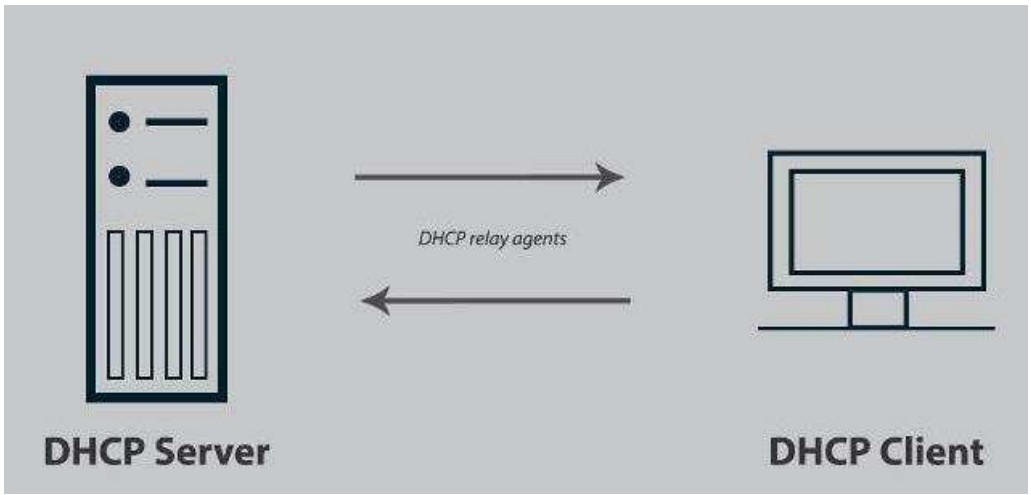
- اجارهٔ یک IP آدرس برای مدت معین به کلاینت‌ها و تمدید دوبارهٔ IPهای اجاره‌یی به‌صورت خودکار برای کلاینت‌هایی که آن را درخواست می‌کنند.
- به‌روزرسانی پارامترهای مربوط به DHCP Clientها توسط DHCP Server و تنظیمات Scope به جای انجام این اعمال به‌صورت دستی بر روی هر کمپیوتر؛

- ریزرف IP آدرس برای دستگاه‌ها و کمپیوترهای معین که همیشه باید همان IP آدرس را داشته باشند.
- Excludeکردن یک رنج مشخص از IP آدرس‌ها برای توزیع بر روی سرورها، روترها و دستگاه‌های دیگر.
- ارائهٔ خدمات به بسیاری از subnetها (اگر روترهایی که بین Dhcp سرور و subnetها هستند تنظیمات مربوط به Agent DHCP را داشته باشند.)
- تنظیمات DHCP سرور برای انجام خدمات DNS به کلاینت‌ها؛
- ارائهٔ آدرس Multicast برای کلاینت‌ها.

۶.۱.۴ کارکرد DHCP

در معماری DHCP سه بخش حائز اهمیت است: یک مشتری DHCP، یک سرور DHCP و عامل رلی DHCP (DHCP relay agents).

مشتری یا کلاینت، هر دستگاهی است که می‌تواند به انترنت وصل شود و با سرور ارتباط برقرار کند. نه‌تنها تلفن‌ها و سیستم‌های کمپیوتری مشتری محسوب می‌شوند، بلکه پرنترها و سرورهای داخل شبکه نیز شامل مشتریان هستند. سرور DHCP یک سیستم کمپیوتری است که کار اختصاص IP را انجام می‌دهد.



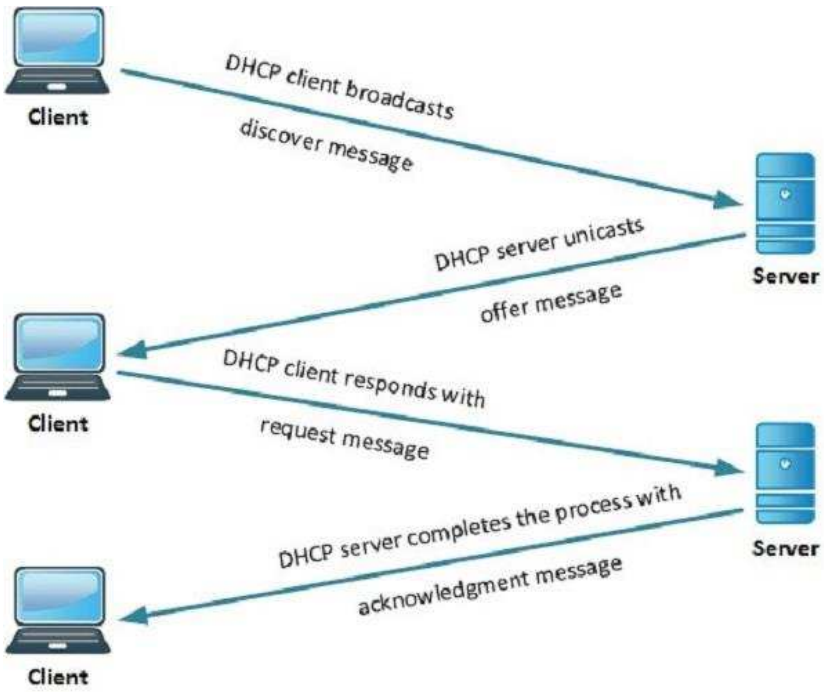
شکل(۶-۴) DHCP relay

DHCP relay agents سیگنال تقاضا بین کلاینت و سرور را انجام می‌دهند. آنها بخش ضروری یک شبکه نیستند، ولی در شبکه‌های عظیم حضور آنها لازم است.

۶.۱.۵ مراحل اجرای درخواست DHCP

وظیفهٔ DHCP ارسال معلومات TCP/IP به سیستم‌های داخل شبکه است؛ معلوماتی از جمله IP ADDRESS, SUBNET MASK, DEFUALT GATEWAY و DNS SERVER بدین منظور سیستم‌ها باید درخواست خود را مبنی بر دریافت IP برای DHCP ارسال کنند تا DHCP یک IP در اختیار آن سیستم قرار دهد.

دریافت IP توسط سیستم‌ها در ۴ مرحله صورت می‌گیرد:



شکل(۶-۵) مراحل درخواست IP از سرور DHCP

مرحلهٔ اول: در این مرحله از طرف کلاینت پیغامی با نام DHCP DISCOVER به سمت DHCP SERVER ارسال می‌شود و کلاینت از سرور تقاضای IP می‌کند.

مرحلهٔ دوم: DHCP SERVER پیغامی با نام DHCP OFFER به سمت کلاینت ارسال می‌کند و در آن پیغام یک IP به کلاینت پیشنهاد می‌دهد؛ مثلاً:

192.168.5.100

مرحلهٔ سوم: درصورتی‌که کلاینت پیشنهاد IP 192.168.5.100 را از طرف سرور بپذیرد پیغام DHCP REQUEST را مبنی بر پذیرفتن این IP به سرور ارسال می‌کند.

مرحلهٔ چهارم: در این مرحله سرور پیغام DHCP REQUEST را از طرف کلاینت قبول می‌کند و آن IP را برای کلاینت در خود ثبت می‌کند.

۶.۱.۶ زمان تخصیص (lease time)

یک آدرس IP از زمانی‌که اختصاص می‌یابد، دارای دورهٔ عمر محدودی است. IP که توسط DHCP امروز به یک سیستم اختصاص می‌یابد، ممکن است متفاوت با فردا باشد. اگر سیستم پیش از این‌که مدت زمان تخصیص (lease time) از میان برود، به شبکه برگردد یا در شبکه بماند، آی‌پی آن تغییر نمی‌کند. در غیر این‌صورت IP جدیدی به دستگاه اختصاص داده می‌شود.

۶.۱.۷ هدف DHCP

وجود DHCP در شبکه برای تشخیص تعداد دستگاه‌هایی که می‌تواند به شبکه متصل باشد، ضروری است. با ورود هر دستگاه به شبکه لازم است فوراً یک آدرس IP به آن اختصاص یابد، طوری‌که با سایر IPهای موجود در شبکه تداخل نداشته باشد. عدم حضور DHCP منجر به تداخل IP و درنتیجه مانع از اتصال دستگاه به شبکه به‌سادگی و سرعت می‌شود. این مسأله یکی از مشکلات مدیریت شبکه است. اختصاص IP به‌صورت دستی و رفع مشکل تداخل حتی در شبکه‌های کوچک، کاری ملال‌آور و زمان‌بر است. در شبکه‌های بزرگ چنین کاری تقریباً غیر ممکن است.

۶.۱.۸ تداخل IP با DHCP

با این‌که DHCP مسئول اختصاص IP است، گاهی می‌تواند خود عامل تداخل IP(IP Confilict) نیز باشد. وجود خطا در DHCP باعث ایجاد این مشکل می‌شود، اما خود این پروتوکول می‌تواند در حین کار مشکل را برطرف کند. اغلب اوقات زمانی‌که خطای تداخل IP را روی سیستم خود می‌بینید، تنها کافیست آن را نادیده بگیرید تا مشکل خود به خود برطرف شود. اگر مشکل باقی بماند، باید روتر را راه‌اندازی (Restart) کنید. باز هم اگر مشکل تداخل برطرف نشود، احتمالاً با مسألهٔ بزرگتری در شبکه روبه‌رو هستید که روتر و DHCP با آن دست به‌گریبان هستند.

هر وسیله در شبکه می‌تواند درخواست تجدید تخصیص DHCP کند. این درخواست موجب ایجاد یک IP آدرس جدید برای وسیله می‌شود. برای این کار از تنظیمات شبکه روی کمپیوتر یا تنظیمات Wifi روی تلفن همراه استفاده می‌شود.

۶.۱.۹ عیارسازی سرور DHCP

حال که با مفاهیم نظری DHCP آشنا شدید در ادامه به نحوهٔ نصب و راه‌اندازی سرویس DHCP در لینوکس خواهیم پرداخت. برای نصب سرویس‌دهندهٔ DHCP به‌صورت زیر عمل می‌کنیم. ابتدا بستهٔ dhcp را نصب می‌کنیم.

```
yum install dhcp
```

در این سناریو فرض بر این است که شما چندین کارت شبکه دارید و قصد دارید روی یک انترفیس مثلاً eth۱ این سرویس را راه‌اندازی نمایید. فایل زیر را با یک ویرایشگر مانند vi یا nano باز کنید:

```
/etc/sysconfig/dhcpd
```

خط زیر را به این فایل اضافه و نتیجه را ذخیره نمایید:

```
DHCPDARGS=eth1
```

به‌صورت پیش‌فرض عیارسازی سرویس‌دهندهٔ DHCP فاقد تنظیمات لازم برای ارایهٔ سرویس به شبکه هست که می‌بایست با ویرایش فایل /etc/dhcp/dhcpd.conf این سرویس را عیارسازی نمود. (یک نمونه از فایل مثال عیارسازی مربوط به این سرویس در مسیر /usr/share/doc/dhcp*/dhcpd.conf.sample موجود است که می‌توانید از آن استفاده کرده ویا مفکوره بگیرید).

```
vi etc/dhcp/dhcpd.conf/
```

نوت: آدرس‌های ارائه‌شده توسط سرویس‌دهنده یا سرور به‌صورت اجارهای leased می‌باشد و پس از اتمام مدت زمان پیش‌فرض یا تعریف‌شده در سرویس‌دهنده، آن آدرس از سرویس‌گیرنده گرفته می‌شود و همچنین به‌دلیل ماهیت dynamic پروتوکول ممکن است آدرسی که زمانی به میزبان شما تخصیص داده شده، به میزبان دیگری در شبکه تخصیص داده شود. برای تغییر مدت زمان پیش‌فرض اجارهٔ (lease) می‌توانیم به‌صورت زیر عمل کنیم.

```
default-lease-time 600;
max-lease-time 7200;
```

عبارت مقابل default-lease-time مدت زمان اجارهٔ پیش‌فرض تنظیمات و max-lease-time حد اکثر زمان اجاره بر حسب ثانیه را مشخص می‌کند. برای تعریف یک Subnet جهت ارایهٔ سرویس در آن می‌بایست مانند مثال زیر تعاریف مربوط را به فایل dhcpd.conf اضافه نمود؛ برای مثال در سرور نمونه:

```
default-lease-time 6000;
max-lease-time 7200;
subnet 192.168.5.0 netmask 255.255.255.0}
range 192.168.5.190 192.168.5.200;
{
```

همچنین توسط عبارت range می‌توانید رنج آدرس‌های IP قابل تخصیص توسط سرویس‌دهنده را مشخص نمایید که در مثال بالا از آدرس 192.168.5.190 الی 192.168.5.200 قابل ارایه در این subnet است. تا این قسمت سرویس‌دهنده برای ارائهٔ آدرس IP عیارسازی شده است. ولی سیستم‌های شبکه علاوه بر آدرس IP نیاز به تنظیمات دیگری نظیر آدرس سرویس‌دهندهٔ DNS و همچنین آدرس Gateway شبکه دارند که برای اضافه‌نمودن آنها به‌صورت زیر عمل می‌کنیم.

```
subnet 192.168.5.0 netmask 255.255.255.0}
range 192.168.5.190 192.168.5.200;
option routers 192.168.5.1;
{
```

عبارت option domain-name-servers آدرس سرویس‌دهندهٔ DNS را مشخص می‌کند و عبارتoption routers آدرس Gateway شبکه را مشخص می‌کند.

```
option domain-name-servers 192.168.5.1
```

شاید شما بخواهید که برای یک میزبان خاص در شبکه (برای مثال: پرنتر شبکه یا دوربین IP ویا سرور شبکه) آدرس IP را ریزرف کنید تا همیشه آن آدرس برای میزبان مورد نظرتان تخصیص داده شود:

```
host printer }
 hardware ethernet 00:16:d3:b7:8f:86;
 fixed-address 192.168.5.195;
{
```

و در آخر سرویس dhcp را فعال و راه‌اندازی می‌کنیم:

```
[root@Forozan1 dhcp]# systemctl enable dhcpd
systemctl start dhcpd
```

۶.۲ عیارسازی FTP سرور

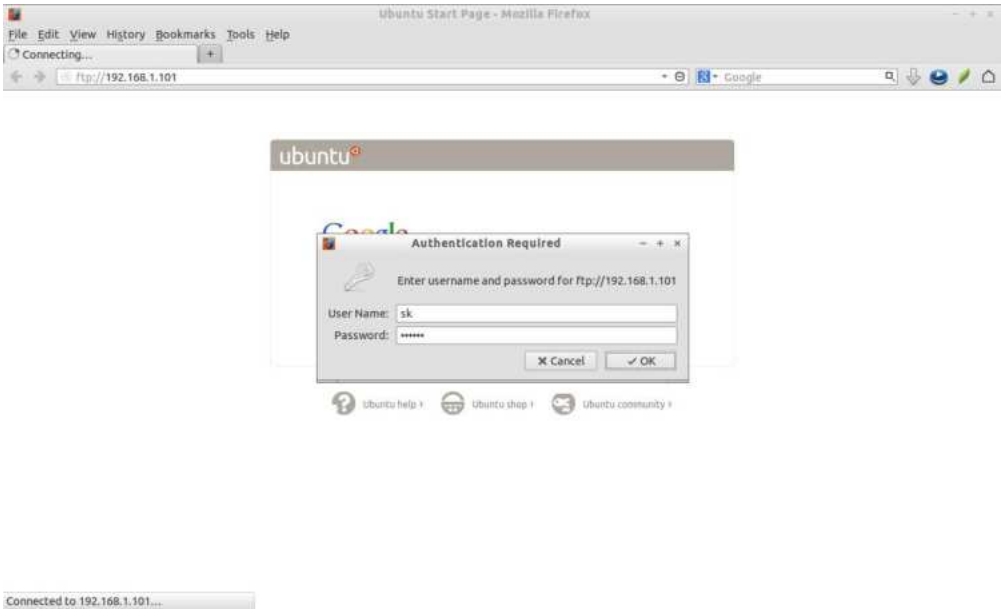
FTP یا به انگلیسی File Transfer Protocol پروتوکولی برای انتقال فایل‌ها بین شبکه‌ها است که اولین بار در سال ۱۹۷۱ جهت انتقال فایل‌ها بین شبکهٔ ArpaNet که متشکل از شبکه‌های دانشگاهی و نظامی بود معرفی شد. شما با استفاده از این پروتوکول می‌توانید فایل‌های خود را روی شبکهٔ داخلی خود ویا سرور خود انتقال دهید.

برای انتقال فایل‌ها بین دو کمپیوتر یا سرور از طریق پروتوکول FTP نیاز است تا روی سیستم مبدأ یا کلاینت از نرم‌افزارهای FTP کلاینت، و روی کمپیوتر یا سرور مقصد FTP سرور نصب و راه‌اندازی گردد. FTP به‌صورت پیش‌فرض و حالت Normal روی پورت TCP 21 اجرا می‌گردد، البته به‌صورت Passive نیز می‌تواند از پورت‌های Dynamic که قابل تعریف در نرم‌افزار FTP سرور است استفاده کند. با استفاده از این پروتوکول شما قادر هستید از امکانات زیر استفاده کنید:

- مشاهدهٔ لست تمام دایرکتری‌های تعریف‌شده در حساب کاربری FTP؛
- تغییر نام فایل و دایرکتری؛
- حذف فایل یا دایرکتری؛
- انتقال فایل و دایرکتری بین دایرکتری‌های مشخص؛
- ایجاد دایرکتری و فایل جدید؛
- انتقال یا آپلود فایل از کمپیوتر مبدأ به سرور یا کمپیوتر مقصد؛
- انتقال یا دانلود فایل از سرور FTP به کمپیوتر مقصد.

۶.۲.۱ آدرس و نحوهٔ دسترسی به FTP

به‌صورت پیش‌فرض با استفاده از ftp:// می‌توانید روی پورت 21 و به‌صورت نورمال به سرور FTP مد نظر متصل شوید. از این آدرس می‌توانید روی مرورگرهای انترنتی استفاده ویا با استفاده از نرم‌افزارهای FTP کلاینت که در ادامه به معرفی آنها خواهیم پرداخت، به سرور FTP متصل گردید؛ برای مثال: اگر سرور 192.168.1.101 یک سرور FTP باشد از طریق آدرس ftp://192.168.1.101 می‌توان به آن دسترسی پیدا کرد.



شکل(۶-۶) اتصال به FTP از طریق مرورگر

۶.۲.۲ نحوهٔ کار FTP

تمامی عملیات در بین کلاینت و سرور در FTP از طریق دستورات از پیش تعریف‌شده پروتوکول FTP انجام می‌گیرد، این کُدها و دستورات ثابت می‌باشند و وضعیت اتصال شما، معلومات تبادله‌شده از طریق آنها انجام می‌گیرد. هنگامی که از کمپیوتر کلاینت یک اتصال FTP ایجاد می‌گردد با استفاده از دستورات مشخص و از پیش تعریف‌شده عملیاتی که در بالا ذکر شد انجام می‌شود؛ به‌عنوان مثال وقتی شما قصد حذف یک یا چند فایل و دایرکتری را روی سرو FTP دارید در ترمینال از دستور rmdir به همراه آدرس فایل یا دایرکتری استفاده می‌کنید.

برخی از دستورات مورد استفاده توسط پروتوکول FTP به شرح زیر می‌باشد:

- Get دانلود فایل یا فایل‌های درخواستی از FTP؛
- Ls جهت مشاهدهٔ لست فایل‌ها و دایرکتری‌ها؛
- Lcd جهت تغییر مکان فولدر فعلی FTP؛
- Mkdir جهت ایجاد یک فولدر روی FTP؛
- Put جهت کپی یک فایل از سیستم مبدأ به FTP سرور؛
- Rmdir جهت حذف یک فولدر در FTP؛
- Quit جهت خروج از FTP.

برای مشاهدهٔ لست تمام دستورات ftp، کافی است علامهٔ ؟ را در ترمینال تایپ کنید و انتر بزنید.

۶.۲.۳ عیارسازی سرور FTP

FTP هم یک سرویس است که باید از طریق نصب یک بستهٔ نرم‌افزاری نصب شود یکی از این بسته‌های نرم‌افزاری vsftpd[۱] می‌باشد، که امنیت آن بهتر است. vsftpd یک سرور FTP امن و سریع برای سیستم‌های Unix/Linux می‌باشد. در این بخش، نحوهٔ تنظیمات یک سرور FTP اولیه را با استفاده از vsftpd بر روی CentOS خواهیم دید. این دستورالعمل همچنین در تمامی نسخه‌های RHEL CentOS، Scientific Linux ۶.x کار خواهد کرد.

در لابراتوار برای کار عملی hostname و IP به‌صورت server.example.local و ۱۹۲.۱۶۸.۱.۱۰۱ می‌باشند. آنها را در سناریو ی خود تغییر دهید.

۶.۲.۴ نصب vsftpd

تمام دستورات با کاربر root اجرا خواهند شد. دستور زیر را در ترمینال برای نصب بستهٔ نرم‌افزاری vsftpd اجرا کنید:

```
# yum install vsftpd ftp -y
```

۶.۲.۵ تنظیم vsftpd

فایل تنظیمات vsftpd را باز کنید. /etc/vsftpd/vsftpd.conf

```
# vi /etc/vsftpd/vsftpd.conf
```

خطوط زیر را پیدا کنید و تغییرات را مطابق آنچه که در کامنت‌ها اشاره شده است انجام دهید:

```
## Set to NO##
anonymous_enable=NO
## Uncomment ##
ascii_upload_enable=YES
ascii_download_enable=YES
## Uncomment - Enter your Welcome message - This is optional ##
ftpd_banner=Welcome to example FTP service.
## Add at the end of this file ##
use_localtime=YES
```

[۱] - Very Secure File Transport Protocol Daemon

سرویس vsftpd را start کنید و آن را در boot سیستم قرار دهید که بعد از هر بار reset شدن سرور به‌طور خودکار اجرا شود:

```
# service vsftpd start
# chkconfig vsftpd on
```

۶.۲.۶ ایجاد کاربران FTP

به‌طور پیش‌فرض کاربر root اجازهٔ وارد به سرور ftp را برای اهداف امنیتی ندارد. بنابراین اجازه دهید کاربری به نام "sk" برای تست با پس‌ورد "centos" ایجاد کنیم.

```
# useradd sk
# passwd sk
```

۶.۲.۷ اتصال به سرور FTP

در این بخش نحوهٔ اتصال از طریق کاربر sk را که در مرحلهٔ قبل ساختیم بیان می‌کنیم، به کُدهای زیر توجه کنید:

```
#ftp 192.168.1.101
Connected to 192.168.1.101 (192.168.1.101).
220 Welcome to example FTP service.
Name (192.168.1.101:root): sk
331 Please specify the password.
Password:
500 OOPS: cannot change directory:/home/sk
Login failed.
ftp>
```

اما با اجرای کُد فوق، شما پیغام خطای زیر را مشاهده می‌کنید:

```
500-OOPS: cannot change directory
```

خطا به این دلیل است که SELinux کاربر را برای ورود به سرور ftp محدود می‌کند و اجازهٔ ورود به کاربر از طریق ftp را به‌طور پیش‌فرض نمی‌دهد. بنابراین باید قیمت Boolean (True & False) در SELinux را برای سرویس FTP تغییر دهیم، تا کاربر اجازهٔ ورود از طریق سرویس FTP را پیدا کند، به کُدهای زیر توجه کنید:

```
# setsebool -P ftp_home_dir on
```

مجدداً برای ورود به سرور FTP تلاش می‌کنیم:

```
# ftp 192.168.1.101
Connected to 192.168.1.101(192.168.1.101).
220 Welcome to example FTP service.
Name (192.168.1.101:root): sk
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

با اجرای کُد فوق خواهید دید که با موفقیت به سرور FTP متصل می‌شوید.

۶.۲.۸ تنظیمات سمت کلاینت

فرض کنید ما یک کلاینت با سیستم‌عامل لینوکس Ubuntu داریم و قصد اتصال به سرور FTP را از طریق این کلاینت داریم، کُد اتصال:

```
$ ftp 192.168.1.101
ftp: connect: No route to host
ftp>
```

شما ممکن است خطایی "ftp:connect:No route to host" را مشاهده کنید. برای حل این مشکل اجازه دهید پورت پیش‌فرض ftp را از طریق فایروال یا روتر باز کنیم. در سمت سرور دستورا ت زیر را انجام دهید. این خطا به دلیل بسته‌بودن پورت ftp از طرف firewall یا Router می‌باشد. اگر firewall لینوکس مانع این کار شده باشد، فایل etc/sysconfig/iptables/ را تغییرات بیاورید، این فایل را باز کرده و تغییرات بیاورید:

```
# vi /etc/sysconfig/iptables
```

خطوط زیر را اضافه کنید.

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 21 -j ACCEPT
```

فایل را ذخیره کرده و خارج شوید. و iptables را restart کنید.

```
# service iptables restart
```

مجدداً از سیستم کلاینت خود برای ورود به سرور FTP تلاش کنید:

```
$ ftp 192.168.1.101
Connected to 192.168.1.101.
220 Welcome to example FTP service.
Name (192.168.1.101:sk): sk
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

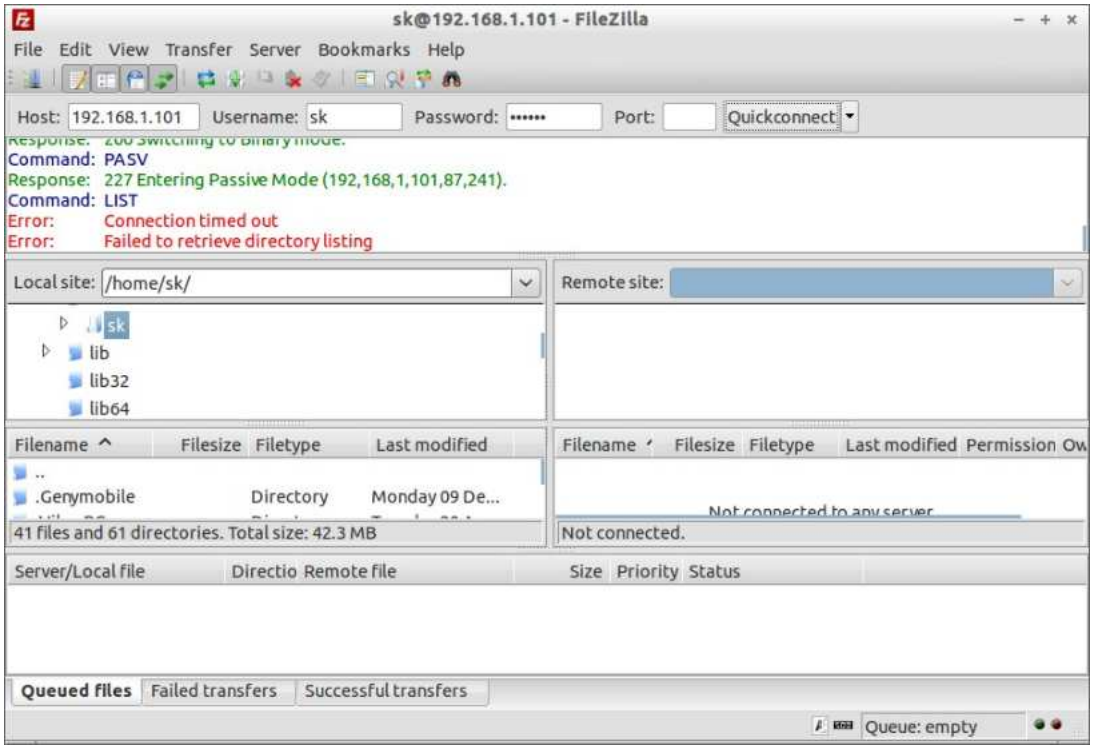
کارکردن از طریق ترمینال ممکن است برای تازه‌کارها دشوار باشد. بنابراین یک نرم‌افزار کلاینت FTP گرافیکی به نام Filezilla را برای انجام راحت‌تر کار با سرور FTP معرفی می‌کنیم، جهت نصب آن کُد زیر را در کلاینت که یک سیستم‌عامل Ubuntu است وارد کنید:

```
$ sudo apt-get install filezilla
```

برای سیستم‌های RHEL، شما می‌توانید filezilla را با استفاده از دستور زیر نصب کنید:

```
# yum install filezilla
```

کلاینت Filezilla را از سیستم خود اجرا کنید. نام سرور FTP خود hostname ویا IP Address، username، password و شمارهٔ پورت را وارد کنید. بر روی "Quickconnect" برای ورود کلیک کنید.



شکل(۶-۷) نرم‌افزار filezilla

ممکن است شما با خطای زیر مواجه شوید.

Error: Connection timed out

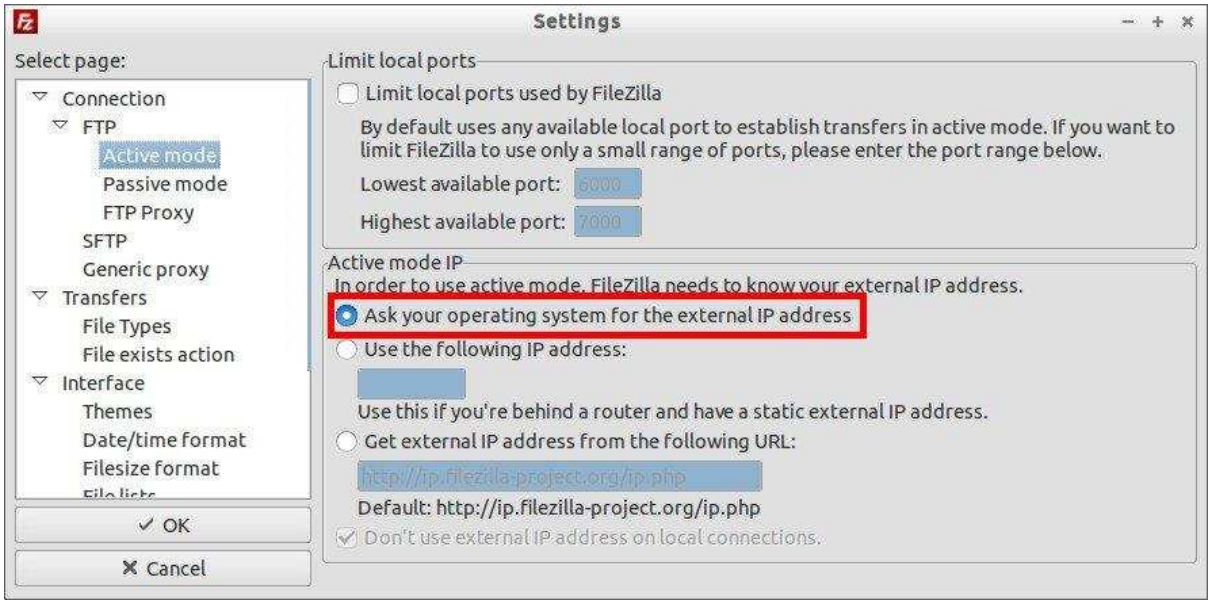
Error: Failed to retrieve directory listing

برای بر طرف‌کردن این خطا، راه حل‌های زیر را دنبال کنید.

راه حل اول

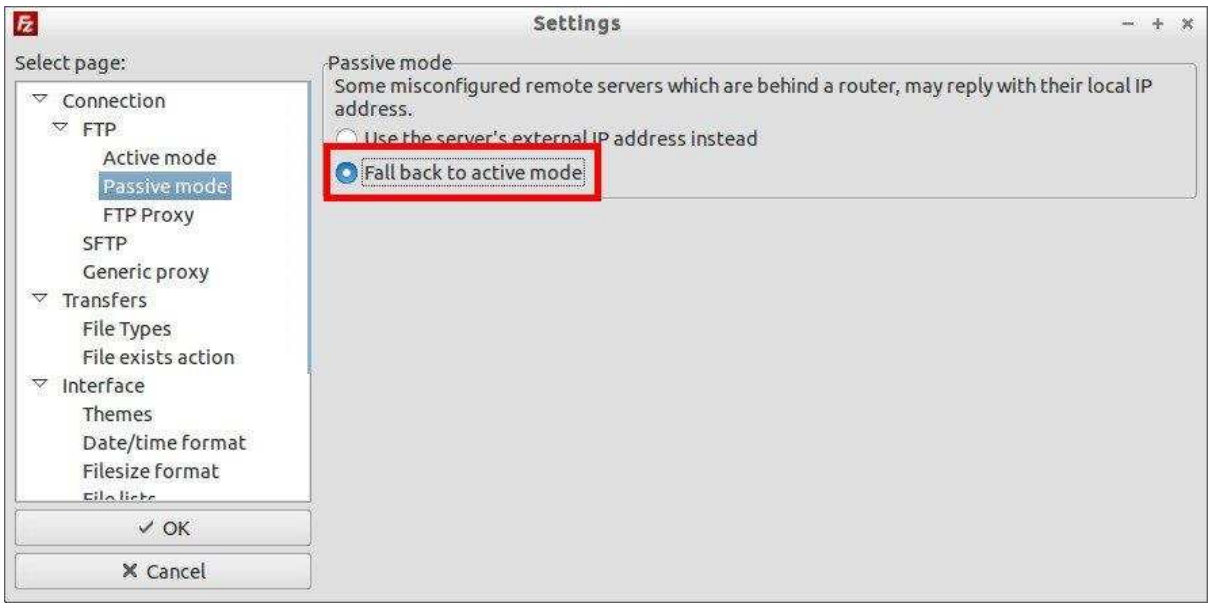
۱. در کلاینت Filezilla خود به قسمت Edit -> Settings -> FTP ->Active Mode بروید.

در تب Active Mode اطمینان حاصل کنید که گزینهٔ “Ask your operating system for the external ip address” انتخاب شده باشد.



شکل(۶-۸) انتخاب گزینهٔ مناسب

سپس به قسمت Passive Mode<- FTP -> Settings -> Edit بروید. گزینهٔ " Fall back to active mode" را انتخاب کنید و بر روی Ok کلیک کنید.



شکل(۶-۹) فعال‌کردن گزینهٔ مناسب دیگر

راه حل دوم

اگر مشکل هنوز حل نشده است، به سرور FTP خود رفته و فایل "/etc/sysconfig/iptables-config" را ویرایش کنید.

```
# vi /etc/sysconfig/iptables-config
```

خط IPTABLES_MODULES را پیدا کنید و آن را برابر با "ip_conntrack_ftp" قرار دهید:

```
[…]
IPTABLES_MODULES="ip_conntrack_ftp"
[…]
```

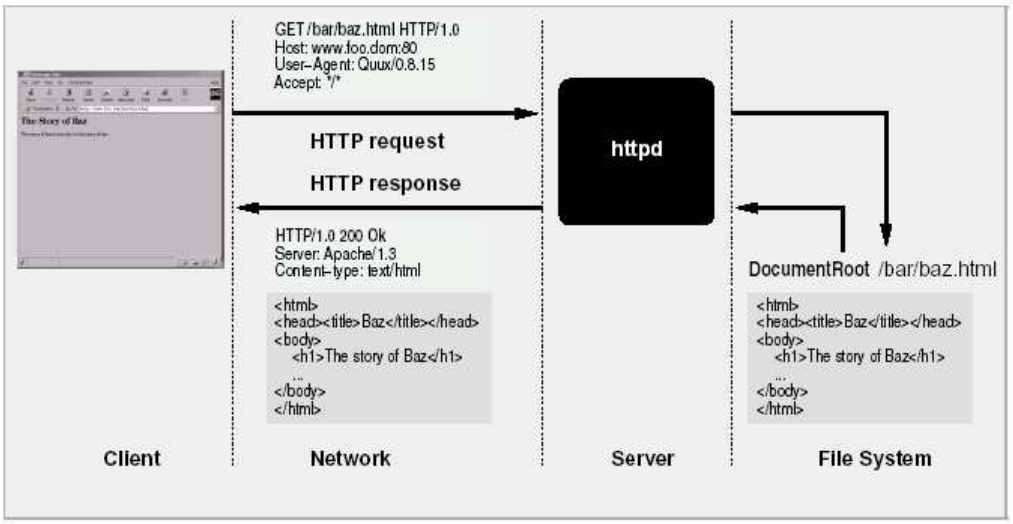
فایل را ذخیره کرده و iptables را restart کنید:

```
# service iptables save
# service iptables restart
```

اکنون دوباره از طریق Filezilla امتحان کنید، مشکل بر طرف خواهد شد.

۶.۳ وب‌سرور

امروزه بسیاری از سرورهای ارائه‌دهندهٔ خدمات میزبانی وب (Hosts) از سیستم‌عامل لینوکس برای ارائهٔ خدمت میزبانی وب‌سایت‌های انترنتی استفاده می‌کنند، چرا که سیستم‌عامل لینوکس و بسیاری از نرم‌افزارهای مبتنی بر آن، متن باز بوده و رایگان می‌باشند و ارائهٔ خدمات میزبانی وب با استفاده از این سیستم‌عامل، هزینهٔ بسیار پایین‌تری داشته و نیازی به خرید لایسنس و محدودیت تحریم وجود ندارد. از این رو بسیاری از وب‌سایت‌های پر مخاطب ایرانی از میزبانی سرورهای لینوکس و نرم‌افزارهای مرتبط با آن استفاده می‌کنند.



شکل(۶-۱۰) نحوهٔ کار یک وب‌سرور

برای ارائه خدمات میزبانی وب در سرور لینوکس، نیاز به راه‌اندازی و استفاده از چند برنامه و نرم‌افزار وجود دارد که در این بخش به آنها خواهیم پرداخت. یکی از این برنامه‌ها که صفحات وب را به کاربر ارائه می‌دهد، وب‌سرور نام دارد. تاکنون برای سیستم‌عامل لینوکس، چندین وب‌سرور ایجاد شده که پر کاربردترین و محبوب‌ترین آن (Apache) است{ Bowen,#۱۶۹۲۰۰}.

در کنار راه‌اندازی وب‌سرور، برای میزبانی وب‌سایت‌های انترنتی، نیاز به ایجاد ساختاری برای ذخیرهٔ معلومات در دیتابیس وجود دارد. از این رو عموماً در سیستم‌عامل لینوکس از دیتابیس MySQL که یکی از قدیمی‌ترین و در عین حال قدرتمندترین دیتابس‌های رابطه‌یی است، استفاده می‌شود. MySQL متن باز بوده و علاوه بر سیستم‌عامل لینوکس در سایر سیستم‌عامل‌ها نیز قابل استفاده است.

قدم آخر، برای این‌که بتوان میزبانی یک وب‌سایت انترنتی را در سیستم‌عامل لینوکس انجام داد، نصب و راه‌اندازی زبان برنامه‌نویسی مورد استفادهٔ وب‌سایت انترنتی است. در سیستم‌عامل لینوکس عموماً از زبان‌های برنامه‌نویسی نظیر: Python, PHP استفاده می‌شود. در این بین، زبان PHP یکی از پرکاربردترین و محبوب‌ترین زبان‌های برنامه‌نویسی برای طراحی وب‌سایت‌های انترنتی است و برای آن چهارچوکات (Frame work)ها و سیستم‌های مدیریت محتوای متعددی طراحی شده است.

هدف از این بخش، آشنایی با LAMP (کوتاه‌شده چهار کلمهٔ PHP، MySQL، Apache، Linux) بوده و در این آموزش، روش نصب و عیارسازی کلی وب‌سرور آپاچی، نرم‌افزار دیتابیس MySQL و زبان برنامه‌نویسی PHP، در توزیع لینوکس اوبونتو (Ubuntu) در محیط KDE و Terminal مورد بررسی قرار می‌گیرد.

در ابتدا انواع مختلف وب‌سرور در لینوکس مورد بررسی قرار گرفته و نحوهٔ نصب و عیارسازی آپاچی، آموزش داده می‌شود. سپس نحوهٔ نصب زبان برنامه‌نویسی PHP۷ آموزش داده شده و فایل تنظیمات آن به‌طور کلی آموزش داده می‌شود. و در پایان، نحوهٔ نصب MySQL و رابط کاربری تحت وب آن phpMyAdmin، مورد بررسی قرار می‌گیرد.

۶.۳.۱ کارکرد وب‌سرور Apache

وب‌سرور آپاچی (Apache) گسترده‌ترین و محبوب‌ترین سرور HTTP در دسترس در انترنت می‌باشد که از زبان‌های PHP و Perl پشتیبانی می‌کند و روی بیشتر سیستم‌عامل‌های قابل اجرا است یک برنامهٔ free Open Source است که با سرورهای وب برای اداره‌کردن درخواست‌ها و تقاضاهای وب و منابع به کار می‌رود.

Apache HTTP Server روی سیستم‌عامل Unix مانند Linux یا BSD اجرا می‌شود همچنین می‌تواند روی Windows مایکروسافت و دیگر سیستم‌عامل‌ها یا Platformها اجرا شود. یک سرور با خصوصیات منحصربه‌فرد با add-onهای قدرتمند که به‌صورت مجانی در دسترس هستند. Apache دارای

امکانات ویژه می‌باشد که متداول‌ترین استفاده از ویژگی‌های این برنامه .htaccess است که طراحان حرفه‌یی در محیط لینوکس از آن بهره می‌گیرند. برای نمونه زمانی‌که بخواهند اولین صفحه در سایت به‌خصوصی باشد با یک دستور در آن پرونده، این امر ممکن می‌گردد ویا زمانی‌که صاحب سایت مایل نیست که فایل‌های موجود در سرور وی توسط دیگران دزدیده شود و بخواهد که مانع از پیوند مستقیم آنها شود Apache کمک می‌کند تا به خواست شان برسند. زمانی‌که برنامه‌نویس بخواهد محل واقعی صفحات دیده نشود، نیز این برنامه مورد استفاده قرار می‌گیرد.

۶.۳.۲ تاریخچه وب‌سرور Apache

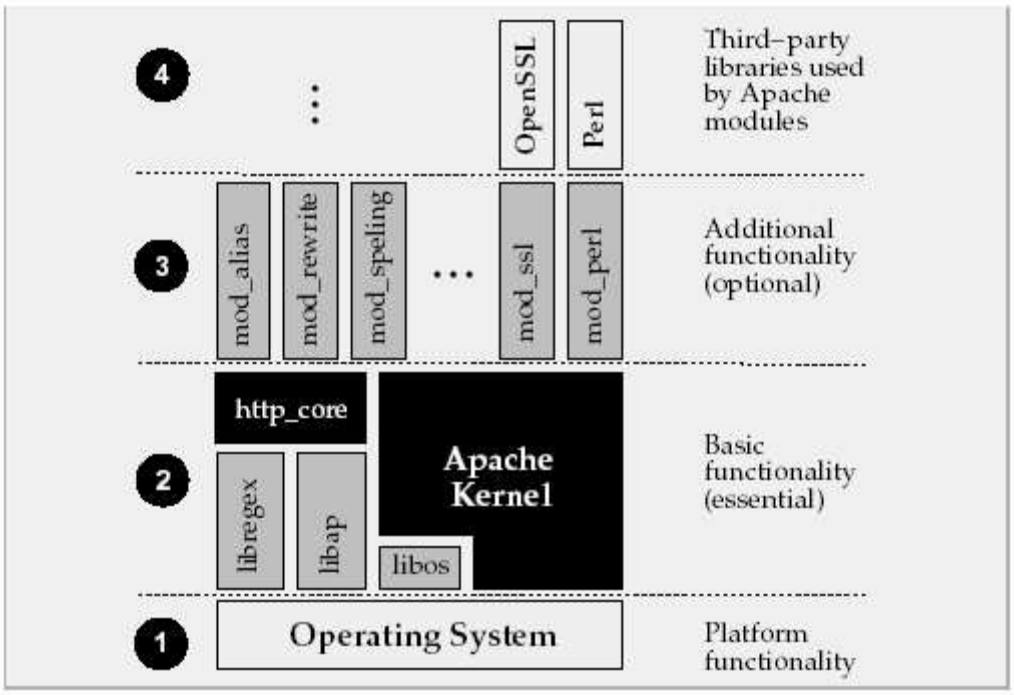
Apache Web Server بیشترین استفاده را روی انترنت دارد بیشتر از ۵۰ فیصد از وب‌سایت‌های موجود از Apache استفاده می‌کنند. این محصول توسط گروهی عرضه می‌شود که به Apache Group موسوم اند و این گروه پروژه Apache Http Server Project را اداره می‌کنند این گروه سخت درکارند تا با ارائهٔ محصولاتی با Configuration بالا و انعطاف‌پذیری خوب عرضه نمایند.

اولین نسخهٔ Apache در April سال ۱۹۹۵ ساخته شد. تا قبل از آپاچی یک وب‌سرور وجود داشت به نام NCSA Httpd 1.3 که در واقع اولین نسخهٔ آپاچی از این وب‌سرو نشأت گرفته است و در واقع پایه و اساس آپاچی بود ولی آپاچی به سرعت رشد کرد و در may-june سال ۱۹۹۵ گروه آپاچی نسخهٔ ۰.۷ را تولید کردند.

نسخهٔ جدید با معماری جدید شامل ساختارهای مادیولر و همچنین بسط توابع API و همچنین تعدادی خصوصیات سیستم‌عامل از قبیل POOL-based memory allocatcon و new forking process model بود و این معماری جدید گسترش یافت تا به آخرین نسخهٔ آن یعنی ۲.۴.۲۳ رسید، البته با ورود این وب‌سرور جدید وب‌سرور قدیمی Ncsa http4 بازنشسته شد و این نسل جدید شروع به کار کرد (Bowen & Coar,۲۰۰۷).

۶.۴ معماری و کارایی وب‌سرور Apache

بستهٔ نرم‌افزاریApache از نظر معماری و کاربرد به چهار لایهٔ زیر تقسیم شده است مطابق با شکل زیر:



شکل(۶-۱۱) معماری وب‌سرور Apache

اساس functionolity در شکل فوق به‌وسیله سیستم‌عامل در لایهٔ زیرین تهیه می‌شود، برای apache سیستم‌عامل مربوطه نسخه‌های گوناگون unix است اما سیستم‌عامل دیگر نیز می‌توانند پاسخ قرار گیرند، سیستم‌عامل‌هایی از قبیل macos.win32.os/2 وحتی posix که سیستم‌عامل کمپیوترهای main frame است.

لایهٔ دوم که هستهٔ اصلی و شامل کرنل و مادیول‌های مربوط به آن و تعدادی librariey استندرد است، این لایه به همراه مادیول‌های مربوطه در عمل Http Server یعنی مبادلهٔ معلوماتی میان Browser و Server مورد استفاده قرار می‌گیرد. این لایه همچنین دارای توابع API و کُدهای قابل استفاده مجدد (REUSABLE) برای لایه‌های بالایی است.

لایهٔ سوم، که لایهٔ مادیول‌های Apache است، لایه‌یی است که آن را از دیگر وب‌سرورها متمایز کرده است و در واقع در این لایه قسمت User-visible Functionality شگفت‌انگیز و تحسین‌برانگیز است و مادیول‌های موجود آنقدر دارای Functionlity بالا هستند که برای سرویس‌دادن استفاده می‌شود و این نشان‌دهندهٔ مستقل‌بودن و وابسته‌نبودن مادیول‌ها نسبت به یکدیگر است. در واقع مادیول‌هایی که مورد استفادهٔ وب‌سرورهای دیگر است. وب‌سرورهای دیگر بدون دسترسی به لایهٔ ۱ و ۲ می‌توانند مستقیم از مادیول‌های لایهٔ ۳ و ۴ استفاده کنند.

در لایهٔ ۳ ممکن است بعضی از مادیول‌ها به تنهایی کارآیی نداشته باشند و برای به‌کاربردن آنها نیاز به libraryهای خارجی است مانند mod-perl و mod-ssl، مثلاً برای استفاده از mod-perl به تعدادی libraryهای زبان perl نیاز داریم. از آنجاکه لایه‌های ۴ و ۳ و مادیول‌های مربوطه مستقل از لایهٔ ۲ هستند و به‌صورت loosely coupled با لایهٔ ۲ هستند و در واقع ارتباط ثابت (Static) با لایهٔ ۲ ندارند و در حقیقت ارتباط آنها با این لایه به‌صورت محرک یا dynamic است و این ارتباط به وسیلهٔ Dynamic Shared Object فراهم می‌شود.

این ویژگی و ساختار دارای انعطاف‌پذیری بالا است، یک نمونه انعطاف‌پذیری این است که به جای این‌که این ارتباط (ارتباط بین ۴و۳ با ۲) در موقع نصب‌کردن آپاچی به‌صورت ثابت (Static) بر قرار شود، این ارتباط در واقع STARTUPTIME لایه‌یی ۴ و ۳ برقرار می‌شود یعنی هر موقع لایهٔ ۳ می‌خواهد با لایهٔ ۲ ارتباط برقرار کند همان موقع پیوند توسط DSO برقرار می‌شود لذا پیوند دایمی نیست. در واقع DSO یکی از ویژگی‌های متمایز آپاچی نسبت به سایر وب‌سرورها می‌باشد.

۶.۴.۱ قابلیت‌های کرنل Apache

هستهٔ آپاچی که در لایهٔ ۲ واقع است دارای دو هدف است:

- تهیهٔ Basic HTTP server functionality (برای مبادلهٔ معلوماتی بین سرور و مرورگر)
- وظایف سیستم‌عاملی از قبیل:
- Memory Segment و... که در واقع Resource Handling می‌شود.
- نگهداری Pre-Forked Process Model
- سرکشی به Socketهای TCP/IP
- کنترول ورود HTTP Request به وسیلهٔ پروسس‌های مربوطه
- مدیریت HTTP Protocol به منظور رسیدگی به در خواست HTTP Request
- تهیهٔ Read/Write Buffer
- تهیهٔ مادیول‌های API و توابع مورد نیاز

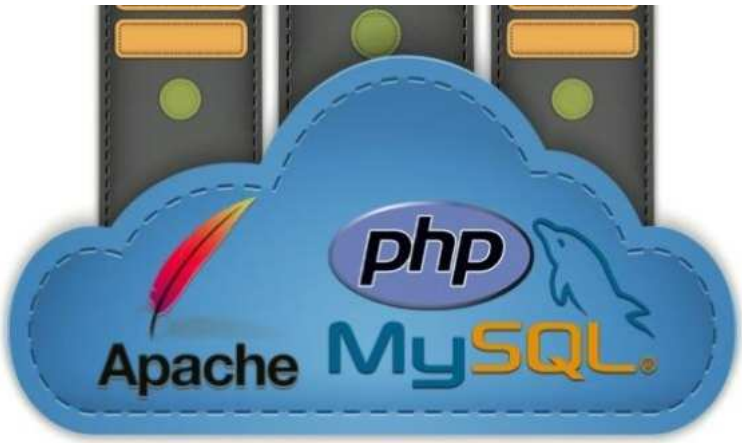
API مادیول‌هایی هستند که مورد استفادهٔ لایهٔ سوم هستند و در هستهٔ آپاچی به وجود می‌آیند، البته این به آن معنا نیست که لایهٔ سوم وابسته به این مادیول‌ها (مادیول‌های API) است. API شامل لست توابعی هستند که در داخل مادیول‌های لایهٔ سوم وجود دارد.

وقتی که هسته یک درخواست HTTP REQUEST دریافت می‌کند برای انتشار این درخواست بین مادیول‌های مختلف از API استفاده می‌کند چرا که API مشخص می‌کند که هر تابع مربوط به چه مادیول است و هر مادیول چه توابعی دارد و هر درخواست پیغام به کدام مادیول باید برود البته API فقط لست توابع موجود در مادیول‌ها نیست بلکه دارای توابعی هست که به‌طور عمومی قابل استفادهٔ مادیول‌های لایهٔ سوم است. این توابع با (ap-xxx) شروع می‌شود هر HTTP REQUEST به ده شاخهٔ مجزا تقسیم می‌شود و

هر مادیول اجرای یکی از شاخه‌ها را بر عهده می‌گیرد و در مواقع لازم نیز از توابع (ap-xxx) استفاده می‌کند (Bowen & Coar,۲۰۰۷).

۶.۴.۲ عیارسازی وب‌سرور

جهت راه‌اندازی یک وب‌سروری که بتواند میزبان یک سایت واقعی باشد، ضرورت به نصب حد اقل سه بستهٔ نرم‌افزاری PHP, MySQL و Apache می‌باشد. به نصب PHP, MySQL و Apache در لینوکس LAMP گفته می‌شود. با نصب سرویس‌های ذکر شده سرور شما آمادهٔ میزبانی از وب‌سایت ویا Application مورد نظرتان می‌شود.



شکل(۶-۱۲) نرم‌افزاری مورد نیاز برای راه اندازیی یک وب‌سرور

نصب LAMP، سرور شما برای میزبانی صفحات HTML، سیستم‌های مدیریت محتوا نظیر WordPress و ... آماده خواهد شد. با توجه به موارد ذکرشده و آگاهی از تمامی ابعاد نصب LAMP، آموزش نصب LAMP روی لینوکس CentOS را در ادامه بیان می‌کنیم.

در این کتاب سعی شده است تا در کنار نصب و عیارسازی اولیه، سرورها به‌طور ساده و عملی تشریح شود. بسته‌های نرم‌افزاری زیر روی لینوکسCentOS باید نصب شود.

- **Apache**
- **MySQL MariaDB**
- **PHP**

برای شروع نصب نیاز به یک سرور با لینوکس CentOS ۷ خواهید داشت. پس دست به کار شده و در سرور خود توضیع ذکرشده را نصب نمایید. سپس از طریق نرم‌افزار Putty یا دیگر کلاینت‌های SSH به سرور متصل شده و مراحل نصب را گام به گام انجام دهید.

۶.۴.۳ نصب Apache

نصب وب‌سرور آپاچی با استفاده از Package Managerها بسیار آسان بوده و تنها با یک دستور می‌توانید آن را روی سرور خود نصب نمایید. برای نصب آپاچی با استفاده از putty به سرور خود متصل شده و دستور زیر را وارد نمایید:

```
yum -y install httpd
```

پس از واردکردن دستور بالا در محیط command-Line وب‌سرور Apache در سرور شما نصب خواهد شد. برای شروع به کار وب‌سرور دستور زیر را در محیط Command-Line وارد نمایید:

```
systemctl start httpd.service
```

تنظیمات firewall برای Apache

شما در این بخش باید در firewall، پورت‌های http و https را (که به شماره‌های ۸۰ و ۴۴۳ می‌باشند) باز کنید. برای این کار باید کُدهای زیر را اجرا نمایید:

```
sudo firewall-cmd --permanent --zone=public --add-service=http
sudo firewall-cmd --permanent --zone=public --add-service=https
sudo firewall-cmd --reload
```

اطمینان از صحت نصب

تا این مرحله Apache در سرور لینوکس centos ما نصب شده است برای اطمینان از صحت نصب و مشاهدهٔ وضعیت Apache کُد زیر را اجرا کنید:

```
sudo systemctl status httpd
```

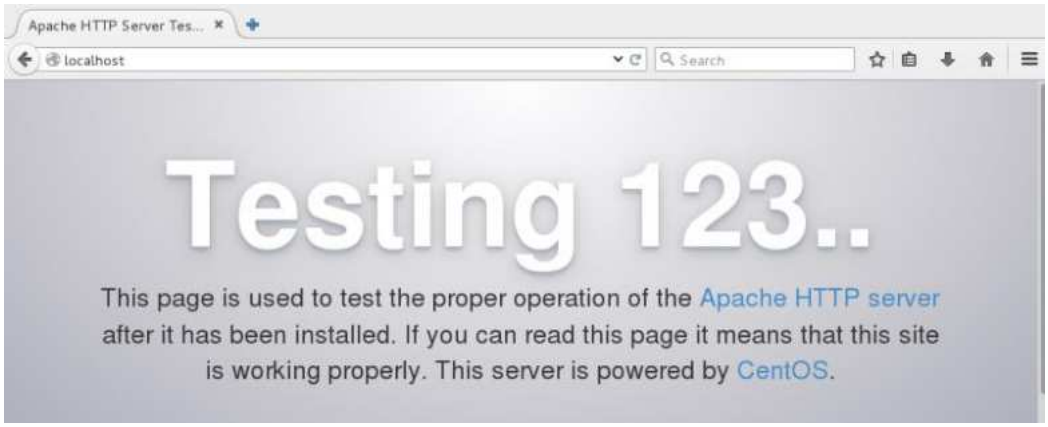
خروجی این دستور باید مشابه زیر باشد:

```
httpd.service - The Apache HTTP Server
Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
Active: active (running) since Thu 2018-04-26 07:13:07 UTC; 11s ago
Docs: man:httpd(8)
man:apachectl(8)
Main PID: 3049 (httpd)
Status: "Total requests: 0; Current requests/sec: 0; Current traffic: 0 B/sec"
CGroup: /system.slice/httpd.service
/ 3049─├ usr/sbin/httpd -DFOREGROUND
/ 3050─├ usr/sbin/httpd -DFOREGROUND
/ 3051─├ usr/sbin/httpd -DFOREGROUND
/ 3052─├ usr/sbin/httpd -DFOREGROUND
/ 3053─├ usr/sbin/httpd -DFOREGROUND
/ 3054─└ usr/sbin/httpd -DFOREGROUND
```

و برای مشاهدهٔ نسخهٔ Apache کُد زیر را اجرا کنید:

```
$ sudo httpd –v
Server version: Apache/2.4.6 (CentOS)
Server built: Oct 19 2017 20:39:16
```

- درنهایت هم برای این‌که مطمئن شوید که وب‌سرور کار می‌کند می‌توانید، مرورگر خود را باز کنید و آدرس IP سرور ویا «دومین» خود را در بخش آدرس وارد کنید و منتظر بمانید تا صفحهٔ وب به شکلی که در تصویر زیر مشاهده می‌کنید، نمایان شود؛ این صفحهٔ پیش‌فرض وب‌سرور Apache است که نشان می‌دهد که وب‌سرور به درستی نصب شده است:



شکل(۶-۱۳) صفحه پیش‌فرض وب‌سرور Apache

۶.۵ مدیریت سرویس‌های Apache

مدیریت سرویس‌های Apache مشابه مدیریت سایر سرویس‌های سیستم‌عامل لینوکس می‌باشد. هدف از مدیریت start, stop, reload, enable , disableکردن سرویس است.

برای توقف سرویس Apache کُد زیر را اجرا کنید:

```
sudo systemctl stop httpd
```

برای شروع سرویس Apache کُد زیر را اجرا کنید:

```
sudo systemctl start httpd
```

برای restartکردن سرویس Apache کُد زیر را اجرا کنید:

```
sudo systemctl restart httpd
```

برای reloadکردن تنظیمات سرویس Apache نیز باید از دستور زیر استفاده کرد، هر زمانی‌که تنظیمات وب‌سرور را تغییر می‌دهید از این دستور برای اعمال تنظیمات می‌توانید استفاده کنید، بدون آنکه سرویس را restart کنید.

```
sudo systemctl reload httpd
```

برای فعال یا غیر فعال‌کردن در وقت boot شدن سیستم‌عامل از دستورات زیر می‌توانید استفاده کنید:

```
sudo systemctl disable httpd
sudo systemctl enable httpd
```

فایل‌های تنظیمات Apache

تمام فایل‌های تنظیمات در دایرکتری (دایرکتری یا folder) /etc/httpd موقعیت دارد.

فایل اصلی تنظیمات /etc/httpd/coonf/httpd.conf است.

تمام فایل های تنظیمات با پسوند conf ختم می‌شوند و فایل های تنظیمات فرعی در آدرس /etc/httpd/conf.d موقعیت دارند و در فای اصلی تنظیمات نیز include می‌شوند.

فا یل تنظیماتی که مسئول loadکردن مادیول های مختلف Apache می باشد، در آدرس /etc/httpd/conf.modules.d موقعیت دارد.

جهت نگهداری آسانتر سرور بهتر است که فایل‌های تنظیمات میزبان‌های مجازی (virtual host) برای هر دومین جداگانه تعریف شوند (میزبان مجازی در عنوان بعدی به‌طور کامل معرفی می‌شوند).

فایل‌های میزبان‌های مجازی باید با پسوند .conf در آدرس /etc/httpd/conf.d تعریف شوند، شما هر تعداد میزبان مجازی می‌توانید برای سرور خود تعریف کنید.

خیلی خوب است که از استندرد نامگذاری برای فایل‌های تنظیمات خود پیروی کنید، برای مثال اگر دومین شما example.com است، پس فال تنظیمات شما باید /etc/httpd/conf.d/example.com.conf نامگذاری شود.

فایل های Apache log (error_log و access_log) در /ver/log/httpd موقعیت دارد. خیلی خوب است که برای هر میزبان مجازی (vhost) خود logهای متفاوت داشته باشید.

شما می‌توانید document root دومین خود را از هر مسیر دیگری انتخاب کنید اما معروفترین مسیرهای معمولاً آین‌ها هستند:

/home/<user_name>/<site_name>

/var/www/<site_name>

/var/www/html/<site_name>

/opt/<site_name>

۶.۶ عیارسازی میزبان مجازی (virtual host configuration)

میزبان مجازی که به آن virtual host گفته می‌شود به وب‌سروری گفته می‌شود که بتوانیم چندین وب‌سایت مختلف را در آن قرار دهیم. به هر یک از وب‌سایت‌ها یک میزبان مجازی گفته می‌شود زیرا از نظر فزیکی همهٔ آنها در یک سرور فزیکی موقعیت دارند. معمولاً هم اکثر وب‌سرورها میزبان چندین سایت انترنتی می‌باشند اما چه طور امکان دارد که همهٔ آنها در یک سرور فزیکی باشند و شما با نام‌های مختلف به یک سرور دسترسی داشته باشید؟ در جواب این سوال باید گفت که این کار تا حد زیادی مربوط به سرور DNS می‌شود که تمام درخواست‌ها را به سمت یک سرور با یک یا چند IP آدرس مشخص راجع می‌کند. در Apache واحد اصلی که یک سایت یا دامنهٔ خاص را تولید می‌کند، یک Virtual Host است. این قابلیت و تقسیمات به مدیران اجازه می‌دهد که با استفاده از میزبان مجازی بتوانند از یک سرور و IP برای میزبانی بیش از یک سایت استفاده کنند.

۶.۶.۱ طریقه ایجاد Virtual Host

در ادامه می‌خواهیم به‌طور عملی طریقهٔ ایجاد دو میزبان مجازی برای دو سایت نمونه به نام‌های example.com , test.com را مرحله به مرحله انجام دهیم.

ایجاد ساختار دایرکتوری

اولین قدم ایجاد یک ساختار دایرکتری برای نگهداری معلومات سایت برای نمایش به بازدیدکنندگان است. دایرکتری root (دایرکتری سطح بالا که آپاچی برای دسترسی به معلومات سایت ابتدا از اینجا شروع به جستجوی معلومات می‌کند) در دایرکتری (آدرس) var/www/ قرار دارد و ما برای ساخت هر میزبان مجازی برای هر سایت در این مکان یک دایرکتری ایجاد می‌کنیم. در هر یک از این دایرکتری‌ها یک زیردایرکتری به نام public_html ایجاد می‌کنیم که محتوای اصلی سایت ما در این مکان قرار می‌گیرد. این مسأله انعطاف‌پذیری بیشتری به میزبان ما می‌دهد؛ به‌عنوان مثال برای سایت‌های معرفی‌شده با این دستور دایرکتری public_html ایجاد می‌کنیم:

```
sudo mkdir -p /var/www/example.com/public_html
sudo mkdir -p /var/www/test.com/public_html
```

گام دوم: دادن مجوز (Permission)

اکنون ما دایرکتوری‌های مورد نیاز را تولید کرده ایم اما تنها توسط کاربر روت می‌توان آنها را مدیریت کرد.

اگر بخواهیم کاربر ما بتواند فایل‌های خود را مدیریت کند باید به آنها دسترسی لازم برای انجام این کار را بدهیم:

```
sudo chown -R $USER:$USER /var/www/example.com/public_html
sudo chown -R $USER:$USER /var/www/test.com/public_html
```

متغیر $User معلومات کاربری است که به سیستم سرور وارد نموده است و هر کاربری از سیستم می‌تواند باشد. با انجام این کار کاربر معمولی ما دارای زیرشاخهٔ public_html است و می‌تواند محتوای سایت را در آن ذخیره کند. همچنین ما باید مجوزهای دسترسی دایرکتری عمومی وب را نیز تغییر دهیم و به آن اجازهٔ خواندن بدهیم.

```
sudo chmod -R 755 /var/www
```

وب‌سرور باید مجوز لازم برای ارائهٔ محتوا و خدمات داشته باشد و همچنین کاربر نیز باید بتواند مطالب را در دایرکتری‌های لازم ایجاد و ذخیره کند.

گام سوم: ایجاد یک صفحهٔ نمایشی برای هر میزبان مجازی

بعد از ایجاد دایرکتری ما نیاز به محتوا برای نمایش به بازدیدکنندگان داریم. حال می‌توانید در این مرحله یک محتوای ساده ایجاد کنید یا اگر محتوای سایت به‌صورت آماده دارید داخل دایرکتری‌ها انتقال دهید و از این مرحله گذر کنید.

با Example.com شروع می‌کنیم و در در داخل دایرکتری public_html آن یک صفحهٔ وب به نام index.html ایجاد می‌کنیم و با ویرایشگر آن را باز می‌کنیم.

زمانی‌که قصد ویرایش یک فایل با دستور nano را دارید، درصورتی‌که فایل مورد نظر موجود نباشد، توسط ویرایشگر nano ایجاد می‌شود و فرصت ویرایش به شما داده می‌شود.

```
nano /var/www/example.com/public_html/index.html
```

معلومات زیر را برای ساخت صفحهٔ نمایشی در آن وارد کنید.

```
<VirtualHost *:80>
    ServerName example.com
    ServerAlias www.example.com
    ServerAdmin webmaster@example.com
    DocumentRoot /var/www/example.com/public_html

    <Directory /var/www/example.com/public_html>
        Options -Indexes +FollowSymLinks
        AllowOverride All
    </Directory>

    ErrorLog /var/log/httpd/example.com-error.log
    CustomLog /var/log/httpd/example.com-access.log combined
</VirtualHost>
```

شکل(۶-۱۴) صفحه وب برای example.com

فایل مورد نظر را ذخیره کنید (Ctrl+O) و ببندید (Ctrl+X).

با استفاده از همین روش برای میزبان دوم نیز یک فایل به همین شکل بسازید. هم در آدرس ویرایشگر و هم در کُدهای HTML به جای Example.com کلمهٔ Test.com را جایگزین نمایید.

گام چهارم: ایجاد فایل‌های اصلی Virtual Host

فایل‌های میزبان مجازی فایل‌هایی هستند که عیارسازی اصلی میزبان مجازی را مشخص می‌کنند به نحوی که وب‌سرور Apache چگونه به درخواست‌های مختلف دومین پاسخ دهد.

Apache یک فایل Vitual Host به نام default.conf-۰۰۰ به‌صورت پیش‌فرض دارد که می‌توانیم برای دیگر Virtual Hostها از آن کاپی و استفاده کنیم.

ما از یک دامنه شروع می‌کنیم، آن را عیارسازی می‌کنیم، سپس آن را برای دیگر دامنه کاپی می‌کنیم و سپس برخی تنظیمات لازم را انجام می‌دهیم.

ایجاد اولین فایل Virtual Host

با کاپی‌کردن فایل برای دامنهٔ اول شروع کنید.

```
sudo cp /etc/apache2/sites-available/000-default.conf /etc/apache2/sites-available/example.com.conf
```

فایل جدید را با دسترسی روت توسط ویرایشگر باز کنید.

```
sudo nano /etc/apache2/sites-available/example.com.conf
```

فایل چیزی شبیه به این خواهد بود. (کامنت‌ها حذف شده تا باعث گیج شدن کاربران نشود.)

```
<VirtualHost *:80>
ServerAdmin webmaster@localhost
DocumentRoot /var/www/html
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

در این بخش تنظیمات و تغییرات را اعمال خواهیم کرد. ابتدا باید ایمل ادمین سرور را تغییر دهیم که ادمین سایت بتواند از طریق آن ایمل‌ها را دریافت کند.

```
ServerAdmin admin@example.com
```

پس از این ۲ دستورالعمل را تعریف می‌کنیم.

اول، نام سرور (Server Name) دومین اصلی را تعریف می‌کند که باید با مقادیر تعریف‌شده در Virtual Host مطابقت داشته باشد. دوم، Server Alias که نام‌های جاگزین دامنه برای میزبان استفاده می‌شود همانند www.

```
ServerName example.com
ServerAlias www.example.com
```

تنها چیزی که ما برای تغییر فایل اصلی Virtual Host نیاز داریم این است که دایرکتری‌یی را که در آن ذخیره شده است بدانیم. و ما فقط نیاز داریم که تنظیمات DocumentRoot را تغییر دهیم.

```
DocumentRoot /var/www/example.com/public_html
```

در مجموع، فایل Virtual Host شما باید بدین صورت باشد.

```
<VirtualHost *:80>
ServerAdmin admin@example.com
ServerName example.com
ServerAlias www.example.com
DocumentRoot /var/www/example.com/public_html
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combine
</VirtualHost>
```

فایل مورد نظر را ذخیره کنید (Ctrl+O) و ببندید (Ctrl+X).

با این دستور فایل تنظیمات را برای دومین دوم کاپی کنید و همین مرحله را برای دامنه دوم تکرار کنید. دقت نمایید به جای example.com از test.com استفاده کنید.

```
sudo cp /etc/apache2/sites-available/example.com.conf /etc/apache2/sites-available/test.com.conf
```

گام پنجم: فعال‌کردن فایل Virtual Host

حال که ما فایل‌های Virtual Host را ایجاد کردیم باید با ابزارهای Apache آنها را فعال کنیم. ما از ابزار ensite۲a برای فعال‌سازی سایت‌ها استفاده می‌کنیم.

```
sudo a2ensite example.com.conf
sudo a2ensite test.com.conf
```

هنگامی که فعال‌سازی پایان یافت، Apache را restart کنید تا تغییرات اعمال شود.

```
sudo service apache2 restart
```

احتمالا پیامی مشابه این پیام دریافت خواهید کرد.

```
Restarting web server apache2
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1. Set the 'ServerName' directive globally to suppress this message
```

این پیام در سایت و کار ما تأثیری ندارد.

گام ششم: امتحان نتیجه

حال که Virtual Host خود را تنظیم کرده‌اید می‌توانید با واردکردن آدرس سایت خود , test.com example.com در مرورگر سایت خود را مشاهده کنید و باید صفحه‌یی شامل این پیام را ببینید.

Success! example.com home page!

درصورتی‌که این تصویر را مشاهده کردید Virtual Host به درستی اعیار شده است.

۶.۶.۲ نصب دیتابیس MySQL [MariaDB]

پس از نصب و راه‌اندازی وب‌سرور در این مرحله باید سرویس دیتابیس را نصب نمایید. برای دیتابیس انتخاب ما MySQL MariaDB است که یک انتخاب مناسب برای سرویس‌دهی دیتابیس می‌باشد. با استفاده از yum می‌توانید به راحتی MySQL MariaDB را نصب نمایید. برای نصب کافیست دستور زیر را در محیط SSH وارد نمایید:

```
yum -y install mariadb-server mariadb
```

پس از نصب با دستور زیر MySQL MariaDB را راه‌اندازی کنید:

```
systemctl start mariadb
```

پس از نصب باید اسکرپت امن‌سازی دیتابیس را اجرا نمایید تا مراحل مربوط را انجام دهید. برای اجرای اسکرِپت امن‌سازی دیتابیس دستور زیر را در دیتابیس وارد نمایید:

```
mysql_secure_installation
```

پس از واردکردن دستور بالا پس‌ورد root از شما خواسته می‌شود، به دلیل تازه‌بودن نصب کافیست بدون واردکردن چیزی Enter را بزنید. در ادامه از شما سوال می‌شود "آیا مایل به تنظیم پس‌ورد برای کاربر root می‌باشید ؟" در صورت تمایل برای قرار دادن کلمهٔ عبور برای کاربر root کلید y را بزنید.

```
Enter current password for root (enter for none):
OK, successfully used password, moving on...
Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorization.
New password: password
Re-enter new password: password
Password updated successfully!
Reloading privilege tables..
... Success!
```

در ادامه سوالات دیگری نیز از شما پرسیده خواهد شد که بهتر است مقادیر پیش‌فرض را انتخاب نمایید. برای این منظور کافیست برای دیگر سوالات Enter را بزنید. درصورتی‌که تمایل دارید تا در هنگام Boot سیستم سرویس MySQL MariaDB شروع به کار کنید، کافیست دستور زیر را وارد نمایید:

```
systemctl enable mariadb.service
```

۶.۶.۳ نصب PHP

برای کامپایل و نمایش کُدهای Dynamic از PHP استفاده می‌شود. نصب این ابزار نیز ساده بوده و به راحتی می‌توانید آن را با استفاده از Package Manager نصب نمایید. برای نصب کافیست دستور زیر را در محیط SSH وارد نمایید:

```
yum -y install php php-mysql
```

پس از واردکردن دستور بالا PHP باید بدون مشکل نصب شود. پس از نصب PHP باید وب‌سرور راه‌اندازی (Restart) شود تا توابع PHP فراخوانی شود. برای Restart وب‌سرور آپاچی دستور زیر را وارد نمایید:

```
systemctl restart httpd.service
```

خلاصهٔ فصل ششم

DHCP یک پروتوکول Server/client است که به‌صورت خودکار به Clientها IP و معلوماتی نظیر Subnet و Default Getaway و DNS را ارائه می‌دهد.

DHCP یکی از سرویس‌هایی است که می‌توان در سیستم‌عامل centos نصب و راه‌اندازی کرد. به‌عنوان یک جزء اختیاری در شبکه محسوب شد و می‌تواند به تمام Clientهای دارای سیستم‌عامل اولیه، یک آدرس IP اختصاص بدهد. همچنین DHCP می‌تواند به‌عنوان یک سرویس در مودم‌های ADSL، روترهای شبکه ویا سویچ‌های لایهٔ 3 اجرا شود.

قبل از این‌که DHCP سرور برای کلاینت‌ها IP فراهم کند باید یک رنج IP برای سرور تعریف شـــود. این رنج به‌عنوان scope شناخته می‌شود.

یک آدرس IP از زمانی‌که اختصاص می‌یابد، دارای دورهٔ عمر محدودی است که به آن lease time گفته می‌شود.

برای راه‌اندازی سرور DHCP باید بسته dhcp را درلینوکس نصب کنید. به صورت پیش‌فرض عیارسازی سرویس‌دهندهٔ DHCP فاقد تنظیمات لازم برای ارایهٔ سرویس به شبکه است که می‌بایست با ویرایش فایل /etc/dhcp/dhcpd.conf این سرویس را عیارسازی نمود. (یک نمونه از فایل مثال عیارسازی مربوط به این سرویس در مسیر /usr/share/doc/dhcp*/dhcpd.conf.sample موجود است که می‌توانید از آن استفاده کرده ویا مفکوره بگیرید).

برای انتقال فایل از طریق FTP در سرور باید FTP سرور نصب باشد و در کلاینت هم FTP Client نصب باشد. به‌طور پیش‌فرض و حالت Normal روی پورت TCP 21 اجرا می‌گردد.

vsftpd یک سرور FTP امن و سریع برای سیستم‌های Unix/Linux می‌باشد. که از طریق نصب و عیارسازی آن می‌توان یک سرور FTP لینوکس ایجاد کرد. فایل تنظیمات این سرویس /etc/vsftpd/vsftpd.conf است. Filezilla نیز یک نرم‌افزار گرافیکی برای راه‌اندازی FTP سرور است.

یکی از این برنامه‌ها که صفحات وب را به کاربر ارائه می‌دهد، وب‌سرور نام دارد. تاکنون برای سیستم‌عامل لینوکس، چندین وب‌سرور ایجاد شده که پر کاربردترین و محبوب‌ترین آن (Apache) است.

جهت راه‌اندازی یک وب‌سروری که بتواند میزبان یک سایت واقعی باشد، ضرورت به نصب حد اقل سه بستهٔ نرم‌افزاری PHP, MySQL و Apache می‌باشد. به نصب PHP, MySQL و Apache در لینوکس LAMP گفته می‌شود.

تمام فایل‌های تنظیمات در دایرکتری (دایرکتری یا folder) /etc/httpd موقعیت دارد.

فایل اصلی تنظیمات /etc/httpd/coonf/httpd.conf است.

تمام فایل‌های تنظیمات با پسوند conf ختم می‌شوند و فایل‌های تنظیمات فرعی در آدرس /etc/httpd/conf.d موقعیت دارند و در فایل اصلی تنظیمات نیز شامل می‌شوند.

سوالات و فعالیت های فصل ششم

۱. پروتوکول bootp را تشریح نمایید.
۲. پروتوکول‌هایی را که در ارتباط با DHCP کار می‌کنند نام ببرید.
۳. دلایل استفاده از DHCP را بیان کنید.
۴. نقش DHCP Relay Agent در روترها چیست؟
۵. مراحل درخواست DHCP را بیان کنید.
۶. پروتوکول FTP از کدام پورت استفاده می‌کند؟
۷. از طریق FTP چه کارهایی را می‌توانیم انجام دهیم؟
۸. شش دستور از دستورات FTP را به همراهکارکرد آنها بیان کنید.
۹. معماری سرور Apache را با رسم دیاگرام آن تشریح نمایید.
۱۰. فایل‌های تنظیمات Apache را نام برده و تشریح کنید که هر کدام برای چه تنظیماتی استفاده می‌شود.

فعالیت ها

۱. برای یک شبکهٔ محلی یک سرور DHCP مطابق رهنمودهای این فصل عیار بسازید و حد اقل دو scope در آن تعریف کنید، یکی برای شبکه LAN و دیگر برای شبکه بیسیم.
۲. از طریق موبایل خود به شبکه متصل شده و منتظر گرفتن تنظیمات از DHCP بمانید. بعد از اتصال تنظیمات موبایل خود را چک کنید تا مطمئن شوید که از scope بیسیم IP دریافت کرده است.
۳. تنظیمات سرور DNS را در سرور DHCP ثبت کنید تا این تنظیمات نیز به کلاینت‌ها داده شود.
۴. یک FTP سرور مطابق رهنمودهای این فصل عیار سازید.
۵. سرور FTP خود را در سرور DNS معرفی بسازید.
۶. از طریق موبایل تلاش کنید به سرور FTP دسترسی پیدا کرده و فایل‌های آن را دانلود کنید.
۷. یک وب‌سرور Apache را به همراه PHP, MySQL در سرور لینوکس خود نصب کنید.

منابع

1. Adelstein, T., & Lubanovic, B. (2007). *Linux system administration*: " O'Reilly Media, Inc.".
2. Blum, R. (2008). *Linux command line and shell scripting bible* (Vol. 481): John Wiley & Sons.
3. Bowen, R., & Coar, K. (2007). Apache Cookbook: Solutions and Examples for Apache Administration: " O'Reilly Media, Inc.".
4. Garrels, M. (2012). *Introduction to Linux*: Tstc Pub.
5. Negus, C. (2010). Linux Bible 2010 Edition: Boot Up to Ubuntu, Fedora, KNOPPIX, Debian, openSUSE, and 13 Other Distributions (Vol. 682): John Wiley & Sons.
6. Palmer, M. (2007). *Guide to UNIX using Linux*: Nelson Education.
7. Petrov, S. (2018). Patch Delivery Infrastructure in SCADA Systems. In.
8. Sobell, M. G. (2014). A Practical Guide to Fedora and Red Hat Enterprise Linux: Pearson Education.
9. Soyinka, W. (2016). Linux Administration: A Beginner's Guide.
10. Ward, B. (2014). How Linux works: What every superuser should know: no starch press.