Linux Professional Institute

LPIC-1

جلسه دهم: آشنایی با ادیتور *vim*، مدیریت کاربران و گروه‌ها و آشنایی با سیستم‌های لاگ در لینوکس

در این جلسه:

ویدئو اول:

- یادآوری سیستم راه‌انداز systemd
- آشنایی با ادیتور *vim*

ویدئو دوم:

- صحبت در مورد حساب‌های کاربری، گروه‌ها، UID و GID
- آشنایی با دستورهای درگیر در ایجاد و مدیریت حساب‌های کاربری و گروه‌ها
- آشنایی با فایل‌های درگیر در تنظیم محیط کاری یک حساب کاربری
- صحبت در مورد syslog و راه‌اندازی یک سرور syslog و ارسال لاگ‌ها به آن

فهرست مطالب

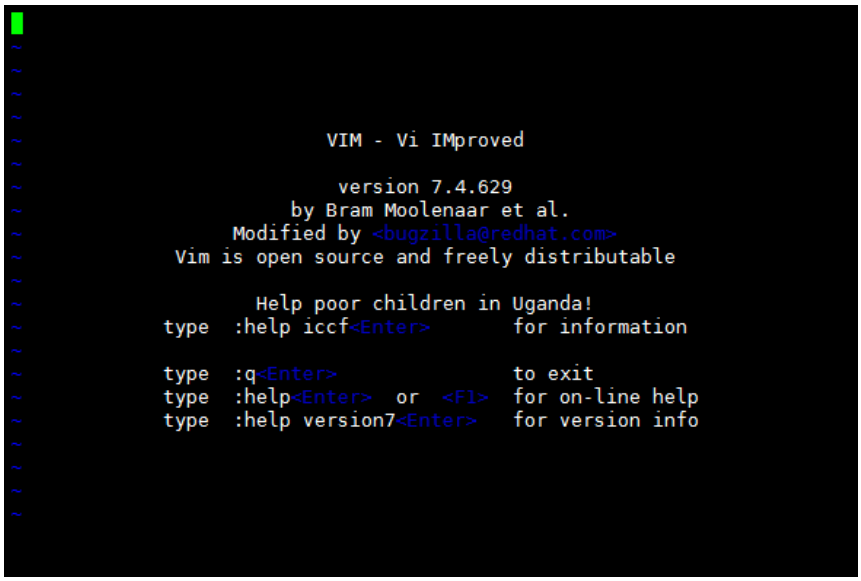
مقدمه

جلسه قبل، فرآیند بوت سیستم را به صورت دقیق بررسی کردیم و سپس به سراغ توضیح در مورد بوت‌لودرها، علملکرد آنها و چگونگی تنظیم آنها رفتیم. پس از آن، در مورد سیستم‌های راه‌انداز صحبت کردیم و با سیستم راه‌انداز SysV و systemd آشنا شدیم و در مورد مدیریت آنها بحث کردیم. در این جلسه، دانش خود از ادیتور vi را کامل‌تر می‌کنیم و سپس به سراغ صحبت در مورد چگونگی ایجاد و مدیریت حساب‌های کاربری می‌رویم. در نهایت، در مورد لاگ‌ها، اهمیت آنها و چگونگی مدیریت آنها صحبت خواهیم کرد.

ایجاد فایل‌های متنی با ادیتور *vi*

vi اولین ادیتوری بود که برای سیستم‌عامل یونیکس نوشته شده بود. این ادیتور طوری طراحی شده بود که هم کوچک و هم ساده باشد؛ به همین دلیل، این ادیتور معمولا به صورت پیش‌فرض روی اکثر سیستم‌های لینوکسی نصب می‌باشد و در شرایط اضطراری (بوت در حالت emergency و...) تنها ادیتوری می‌باشد که در دسترس ما قرار دارد. این امر باعث می‌شود که یادگیری این ادیتور بسیار پراهمیت باشد. ما در جلسه‌ی دوم با چگونگی ایجاد و تغییر فایل‌های متنی به کمک ادیتور vi آشنا شدیم و تا به اینجا نیز از آن برای ایجاد تغییرات در فایل‌های تنظیمات متفاوت، استفاده کرده‌ایم، اما ادیتور vi، قابلیت‌های بسیار زیادی دارد که ما تا به اینجا به آنها نپرداخته‌ایم.

اگر به خاطر داشته باشید، برای استفاده از این ادیتور، کافی است دستور vi را وارد کنیم. به محض وارد کردن این دستور، با نمایی نظیر تصویر ۱ مواجه می‌شویم. اما تا به حال فکر کرده‌اید که چرا در صفحه‌ای اصلی این ادیتور، عبارت VIM نوشته شده است؟



تصویر ۱ - صفحه‌ی اصلی ادیتور vi

تفاوت *vi* با *vim*

همانطور که گفتیم، vi اولین ادیتوری بود که در برای سیستم‌عامل یونیکس نوشته شده بود. این ادیتور تا سال ۲۰۰۲ به راحتی و صورت متن‌باز برای سیستم‌های لینوکسی در دسترس نبود، به همین دلیل بسیاری از کاربران بر آن شدند که یک ادیتور مشابه به vi را به صورت متن‌باز ایجاد کنند. یکی از معروف‌ترین این ادیتورها،

vim، مخفف Vi IMproved می‌باشد. vim ادیتوری است که توسط Bram Moolenaar به عنوان نسخه‌ی بازنویسی شده vi، با بسیاری از امکانات اضافه‌تر، نوشته شده است.

در بسیاری از توزیع‌های لینوکسی، وارد کردن دستور vi، ادیتور vim را برای ما باز می‌کند؛ یعنی در این توزیع‌ها معمولا دستور vi، به عنوان یک Alias برای vim کار می‌کند. در برخی از توزیع‌ها نظیر CentOS، دستور vi، یک نسخه‌ی سبک‌تر از ادیتور vim را برای ما باز می‌کند. این نسخه‌ی سبک‌تر، قابلیت‌های نظیر Syntax Highlighting و... را ندارد ونزدیک‌ترین ادیتور به vi اصلی می‌باشد. در چنین سیستم‌هایی، ممکن است نسخه‌ی کامل vim به صورت پیش‌فرض روی سیستم نصب نباشد و ما مجبور با نصب آن با استفاده از یکی از پکیج منیجرها باشیم (yum و...).

پس به طور خلاصه:

- نسخه‌ی اصلی vi که برای یونیکس نوشته شده بود، در اکثر توزیع‌های لینوکسی وجود ندارد.
- در اکثر توزیع‌های لینوکسی، ادیتور vim که نسخه‌ی بازنویسی‌شده vi اصلی می‌باشد نصب شده است. این نسخه قابلیت‌های بیشتری نسبت به vi اصلی دارد.
- دستور vi در اکثر توزیع‌های لینوکسی، یا یک نسخه‌ی سبک از vim را اجرا می‌کند و یا یک Alias برای خود vim می‌باشد.

در هر حال، چه از نسخه‌ی سبک vim و چه از نسخه‌ی کامل vim استفاده کنیم، مفاهیم اولیه‌ی این ادیتورها بسیار یکسان می‌باشند. بنابراین ما از اینجا به بعد برای سادگی کار، به این ادیتور، vi می‌گوییم.

آشنایی با حالت‌های (modeهای) *vi*

vi همیشه در یکی از سه حالت زیر قرار دارد:

- Command Mode

وقتی vi را باز می‌کنیم در این حالت قرار داریم. بعضا به این حالت Normal Mode نیز می‌گویند. وقتی در این حالت هستیم، می‌توانیم با زدن دکمه‌های کیبورد، یک سری فرمان به vi بدهیم. مثلا در این حالت، با فشردن دکمه‌ی j، مکان‌نما (Cursor) یک خط پایین می‌رود. از این حالت برای انجام کارهایی نظیر Copy، Paste، پاک‌کردن خط‌ها و... استفاده می‌کنیم.

- Insert Mode

حالتی است که در آن می‌توانیم موارد متفاوتی را درون یک فایل تایپ کنیم. در این حالت، vi شبیه به ادیتورهایی نظیر Notepad عمل می‌کند، چرا که در این حالت فقط می‌توانیم متن مورد نظر خود را تایپ کنیم. ما با زدن دکمه‌ی i در Command Mode، به این حالت می‌رویم. به محض ورود به این حالت، در پایین صفحه با نوشته‌ی --Insert-- مواجه می‌شویم. با زدن دکمه‌ی Esc، از این حالت، به Command Mode باز می‌گردیم. بعضا به Insert Mode، Edit Mode یا Entry Mode نیز می‌گویند.

- Ex Mode

حالتی است که در آن می‌توانیم یک سری دستور به vi بدهیم. ما با زدن دکمه‌ی دونقطه (:) در Command Mode، وارد این حالت می‌شویم. دستورهایی که در این حالت به vi می‌دهیم، باید پس از علامت دونقطه قرار گیرند. بعضا به همین دلیل، به حالت Ex Mode، حالت Colon Commands

(دونقطه-دستور) نیز می‌گویند. برای مثال، اگر بخواهیم از vi بدون ذخیره‌ی مواردی که نوشته‌ایم خارج شویم، از q : استفاده می‌کنیم.

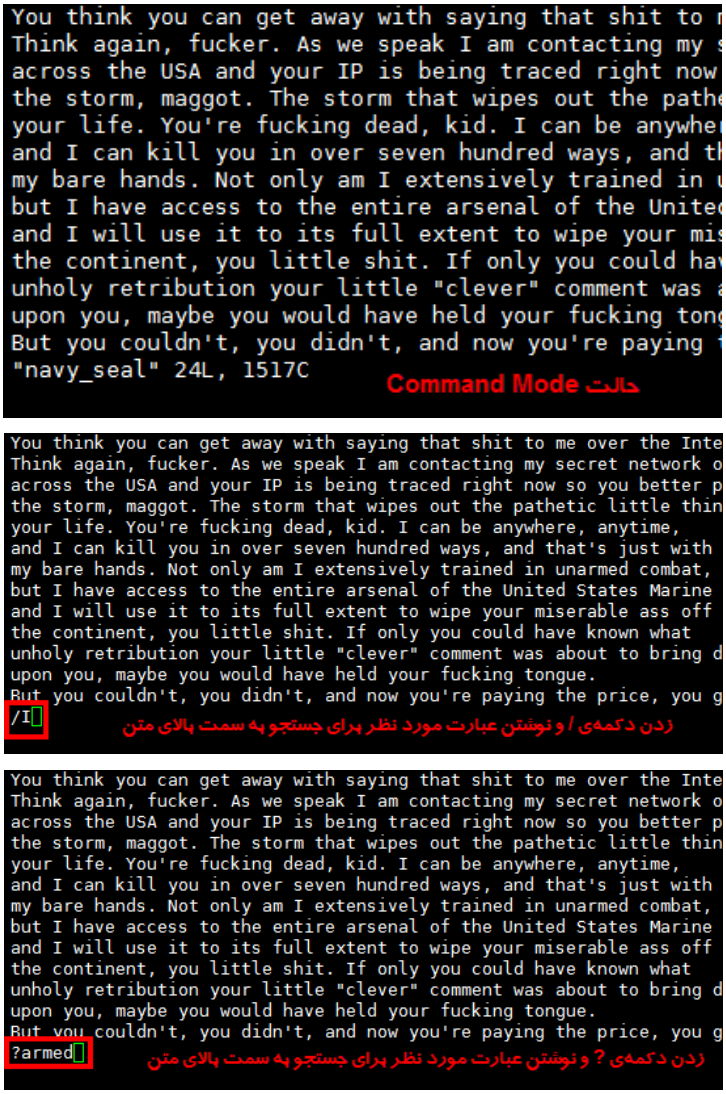
بررسی عملگرهای ویرایش متن

با توجه به این که هنگام باز کردن یک فایل با vi، همیشه در Command Mode قرار می‌گیریم، بهتر است ابتدا با برخی از فرمان‌های موجود در Command Mode برای حرکت در متن آشنا شویم. در جدول زیر، پرکاربردترین فرمان‌ها برای حرکت در یک فایل متنی را مشاهده می‌کنید. توجه کنید که حروف مشخص شده در این جدول را باید دقیقا همانطور که این حروف را تایپ می‌کنیم، وارد کنیم (برای h، فقط دکمه‌ی h کیبورد را فشار می‌دهیم. برای G، دکمه‌ی Shift به علاوه‌ی دکمه‌ی g کیبورد را فشار می‌دهیم).

جدول ۱- پرکاربردترین فرمان‌های Command Mode برای حرکت در یک فایل متنی

کلید	عملکرد
h	مکان‌نما (Cursor) را به اندازه‌ی یک واحد به چپ حرکت می‌دهد.
l	مکان‌نما را به اندازه‌ی یک واحد به راست حرکت می‌دهد.
j	مکان‌نما را به خط بعدی می‌برد.
k	مکان‌نما را به خط قبلی می‌برد.
w	مکان‌نما را روی اولین کاراکتر کلمه‌ی بعدی قرار می‌دهد.
e	مکان‌نما را روی آخرین کاراکتر کلمه‌ی کنونی قرار می‌دهد. فشردن دوباره‌ی آن، مکان‌نما را روی آخرین کاراکتر کلمه‌ی بعدی می‌برد.
b	مکان‌نما را روی اولین کاراکتر کلمه‌ی کنونی قرار می‌دهد. فشردن دوباره‌ی آن، مکان‌نما را روی اولین کاراکتر کلمه‌ی قبلی می‌برد.
^	مکان‌نما را به ابتدای خط کنونی می‌برد.
$	مکان‌نما را به انتهای خط کنونی می‌برد.
gg	مکان‌نما را به ابتدای اولین خط موجود در فایل می‌برد.
G	مکان‌نما را به انتهای آخرین خط موجود در فایل می‌برد.
***n*G**	مکان‌نما را به ابتدای خط شماره‌ی *n* می‌برد.
Ctrl+B	نمای کنونی را تقریبا به اندازه‌ی یک صفحه (صفحه‌ی کنونی ترمینال) به سمت بالا حرکت می‌دهد.
Ctrl+F	نمای کنونی را تقریبا به اندازه‌ی یک صفحه (صفحه‌ی کنونی ترمینال) به سمت پایین حرکت می‌دهد.
Ctrl+U	نمای کنونی را به اندازه‌ی نیم‌صفحه (صفحه‌ی کنونی ترمینال) به سمت بالا حرکت می‌دهد.
Ctrl+D	نمای کنونی را به اندازه‌ی نیم‌صفحه (صفحه‌ی کنونی ترمینال) به سمت پایین حرکت می‌دهد.
Ctrl+Y	نمای کنونی را به اندازه‌ی یک خط به سمت بالا حرکت می‌دهد.
Ctrl+E	نمای کنونی را به اندازه‌ی یک خط به سمت پایین حرکت می‌دهد.

ما می‌توانیم در Command Mode برای یک کلمه یا عبارت نیز جستجو کنیم. برای این کار کافی است دکمه‌ی ? را برای جستجو به سمت جلو و دکمه‌ی / را برای جستجو به سمت عقب فشار دهیم. به محض فشردن یکی از این دو کلید، علامت کلید فشرده شده در پایین صفحه‌ی vi به ما نشان داده می‌شود و ما می‌توانیم کلمه یا عبارت مورد نظر را در جلوی آن تایپ کنیم و سپس دکمه‌ی Enter را برای انجام جستجو فشار دهیم. اگر اولین کلمه یا عبارت پیدا شده کلمه‌ی مورد نظرما نباشد، می‌توانیم بار دیگر دکمه‌ی Enter را بزنیم تا به کلمه‌ی بعدی که با عبارت جستجو شده تطابق دارد برویم.



تصویر ۲- جستجو در یک متن با vi

حال که با چگونگی حرکت در یک فایل متنی آشنا شدیم، وقت آن رسیده که با فرمان‌های ویراش متن در Command Mode نیز آشنا شویم. در جدول ۲، پرکاربردترین فرمان‌های ویرایش متن در Command Mode را مشاهده می‌کنیم. توجه کنید که حروف مشخص شده در این جدول را باید دقیقا همانطور که این حروف را تایپ می‌کنیم، وارد کنیم (برای a، فقط دکمه‌ی a کیبورد را فشار می‌دهیم. برای A، دکمه‌ی Shift به علاوه‌ی دکمه‌ی a کیبورد را فشار می‌دهیم). لازم به ذکر است که با فشردن دکمه‌هایی که در مقابل عملکردشان علامت * وجود دارد، به Insert Mode می‌رویم و برای بازگشت به Command Mode، باید دکمه‌ی Esc را فشار دهیم.

جدول ۲- پرکاربردترین فرمان‌های Command Mode برای ویرایش یک فایل متنی

کلید	عملکرد
a	امکان اضافه‌کردن متن پس از مکان‌نما را به ما می‌دهد.*
A	امکان اضافه کردن متن در پایان خط کنونی را به ما می‌دهد.*
i	امکان اضافه کردن متن قبل از مکان‌نما را به ما می‌دهد.*
I	امکان اضافه کردن متن در ابتدای خط کنونی را به ما می‌دهد.*
o	یک خط جدید در پایین خط کنونی ایجاد می‌کند.*
O	یک خط جدید در بالای خط کنونی ایجاد می‌کند.*
dd	خط کنونی را پاک می‌کند.
dw	کلمه‌ای که مکان‌نما روی آن قرار دارد را پاک می‌کند.
yw	کلمه‌ی کنونی را کپی می‌کند.
yy	خط کنونی را کپی می‌کند.
p	متن کپی شده را پس از مکان‌نما Paste می‌کند.
P	متن کپی شده را قبل از مکان‌نما Paste می‌کند.

نکته: بسیاری از اوقات با استفاده از دکمه‌های جهتی در Command Mode، به موقعیت مورد نظر خود در فایل می‌رویم و سپس با فشردن دکمه‌ی *i*، شروع به اضافه کردن نوشته‌ی خود می‌کنیم. بدین شکل، مجبور به حفظ کردن بسیاری از فرمان‌های مورد استفاده برای ویرایش فایل متنی نخواهیم بود.

Ex Mode نیز چندین فرمان به‌دردبخودر دارد که می‌تواند ما را در انجام کارها یاری دهد. در جدول زیر، پرکاربردترین فرمان‌های Ex Mode که ما را در مدیریت فایل‌ها یاری می‌دهند را مشاهده می‌کنید. همانطور که می‌بینید، همه‌ی این فرمان‌ها با کاراکتر دونقطه (:) شروع می‌شود. در واقع علامت : ما را به Ex Mode می‌برد:

جدول ۳- پرکاربردترین فرمان‌های Ex Mode برای مدیریت فایل‌ها

کلید	عملکرد
:! *command*	بدون خروج از ادیتور، دستور *command* را در شِل سیستم‌عامل اجرا می‌کند و خروجی آن را در یک صفحه‌ی دیگر به ما نشان می‌دهد. با زدن دکمه‌ی Enter به صفحه‌ی ادیتور باز می‌گردیم.
:r! *command*	دستور *command* را در شِل سیستم‌عامل اجرا می‌کند و خروجی آن را درون خود ادیتور، در پایین مکان‌نما، قرار می‌دهد.
:r *file*	*file* را می‌خواند و محتویات آن را درون خود ادیتور، در خط پایین مکان‌نما، قرار می‌دهد.

نکته: توجه کنید که برای رفتن به Ex Mode، باید در Command Mode قرار داشته باشیم؛ اگر در Insert Mode باشیم، کافی است دکمه‌ی Esc را زده تا به Command Mode بازگردیم.

ذخیره‌ی تغییرات و (یا) خروج از ادیتور

یکی از کارهایی که ممکن است بسیاری از افرادی که با ادیتور vi آشنایی ندارند را سردرگم کند، چگونگی خروج از ادیتور و همچنین ذخیره‌ی تغییرات می‌باشد. فرمان‌های زیادی برای خروج و ذخیره‌ی تغییرات در ادیتور vi وجود دارد. در جدول ٤، معمول‌ترین روش‌ها برای ذخیره‌ی تغییرات را مشاهده می‌کنید:

جدول ٤- معمول‌ترین روش‌ها برای خروج و یا ذخیره‌ی تغییرات

Mode	کلید	عملکرد
Ex	`:x`	متن موجود در صفحه را داخل فایل ذخیره کرده و از ادیتور خارج می‌شود.
Ex	`:wq`	متن موجود در صفحه را داخل فایل ذخیره کرده و از ادیتور خارج می‌شود.
Ex	`:wq!`	متن موجود در صفحه را داخل فایل ذخیره کرده و از ادیتور خارج می‌شود (به صورت forced، یعنی اگر برنامه‌ی دیگر در حال کار روی فایل باشد و...، باز هم عمل ذخیره انجام می‌شود).
Ex	`:w`	متن موجود در صفحه را در فایل ذخیره می‌کند، اما از ادیتور خارج نمی‌شود.
Ex	`:w!`	متن موجود در صفحه را در فایل ذخیره می‌کند، اما از ادیتور خارج نمی‌شود (به صورت forced).
Ex	`:q`	از ادیتور، بدون ذخیره‌ی محتویات صفحه در متن، خارج می‌شود.
Command	ZZ	متن موجود در صفحه را داخل فایل ذخیره کرده و از ادیتور خارج می‌شود.

نکته: در صورت وارد کردن دستور vi به تنهایی (یعنی عدم ارائه‌ی نام فایل برای ویرایش، یا نام برای ایجاد فایل جدید)، فرمان‌های `:x`، `:wq`، `:wq!`، `:w`، `:w!` و ZZ کار نخواهند کرد و ادیتور به ما اطلاع می‌دهد که نامی برای فایل ایجاد شده، انتخاب نکرده‌ایم. برای حل این مشکل، کافی است پس از وارد کردن فرمان مورد نظر در Ex Mode، نام مورد نظر برای این فایل جدید را بنویسیم (مثلا `:w sample.txt`).

همانطور که تا اینجا دیدیم، کار کردن با ادیتور vi می‌تواند کمی دشوار باشد. به طور کلی، مواردی که در جلسه‌ی دوم در مورد ادیتور vi گفتیم برای انجام اکثر کارهای ساده کافی می‌باشد، اما به محض این که بخواهیم کارهای دشوارتری را انجام دهیم، باید با ویژگی‌های پیشرفته‌تر vi نیز آشنا باشیم.

نکته: اگر مواردی که در این بخش مشاهده کردید به نظرتان دشوار بود، می‌توانید از برنامه‌ی vimtutor استفاده کنید. این برنامه به صورت پیش‌فرض در بسیاری از توزیع‌ها موجود می‌باشد و سعی می‌کند به صورت Interactive چگونگی کار با ادیتور vi را به ما یاد دهد.

آشنایی با حساب‌های کاربری و گروه‌ها

همه‌ی ما به طور کلی با عملکرد حساب‌های کاربری (User Account) در یک سیستم‌عامل آشنایی داریم. به طور کلی، حساب‌های کاربری در لینوکس دقیقا مانند حساب‌های کاربری در ویندوز یا بسیاری از وبسایت‌ها می‌باشند. در واقع حساب‌های کاربری شناسه‌هایی هستند که ما از طریق آن، خودمان را به سیستم معرفی کنیم و در نتیجه‌ی آن، می‌توانیم از منابع سیستم استفاده کنیم. سیستم نیز با نگاه کردن به اطلاعات حساب کاربری ما،

سعی به تصدیق هویت ما می‌کند و در صورت تصدیق، منابعی که مجوز دسترسی به آن داریم را در اختیار ما قرار می‌دهد.

پرواضح است که به عنوان ادمین سیستم، ما باید با حساب‌های کاربری و چگونگی مدیریت آن توسط لینوکس آشنایی داشته باشیم. ما در این بخش، می‌خواهیم با مفاهیم مربوط به حساب‌های کاربری و گروه‌ها، آشنا شویم.

آشنایی با حساب‌های کاربری در لینوکس

به طور کلی، ما در لینوکس سه نوع حساب کاربری داریم:

- حساب کاربری افراد (حساب‌های کاربری معمولی)

معمولا هر فردی که می‌خواهد به سیستم دسترسی داشته باشد، باید یک حساب کاربری مجزا برای خود داشته باشد. هر حساب‌های کاربری، یک یوزرنیم، معمولا یک رمز و حداقل یک گروه دارد.

- حساب کاربری سرویس‌ها

در لینوکس، برخی از سرویس‌ها (یا daemonها) یک حساب کاربری مخصوص به خود دارد. همانطور که قبلا گفتیم، سرویس‌ها یا daemonها برنامه‌هایی هستند که یک سرویس خاصی را به ما ارائه می‌دهند. daemonها دائما در پشت‌صحنه‌ی سیستم در حال اجرا می‌باشند و گوش به زنگ رویدادی هستند که آنها را برای کار خاصی صدا کند.

نکته‌ی جالب در مورد حساب‌های کاربری مربوط به سرویس‌ها، این است که این حساب‌ها قابلیت وارد شدن (login) به سیستم را ندارند. سرویس‌ها از حساب کاربری خود، برای استارت سرویس‌های جانبی، نوشتن logها و... استفاده می‌کنند. به عبارت دیگر، حساب کاربری یک سرویس، عملکرد و اجازه‌های دسترسی آن سرویس را تحت کنترل نگه میدارد و از نظر امنیتی، سیستم را در مصونیت بالاتری قرار می‌دهد.

- حساب‌های کاربری ویژه

حساب‌های کاربری هستند که برای یک عمل خاص ایجاد می‌شوند. مثلا ممکن است یک حساب کاربری برای دریافت ایمیل داشته باشیم، اما آن حساب کاربری اجازه‌ی ورود به سیستم نداشته باشد.

در لینوکس، هر حساب کاربری باید یک یوزرنیم منحصر به فرد داشته باشد. به طور کلی، لینوکس محدودیت خاصی در مورد چگونگی نام‌گذاری یوزرنیم‌ها ندارد، اما پیشنهاد می‌شود که یوزرنیم‌ها، همیشه از حروف کوچک تشکیل شده باشند. لینوکس به حروف بزرگ و کوچک در یوزرنیم‌ها، حساس می‌باشد؛ یعنی برای لینوکس یوزرنیم behnam با یوزرنیم Behnam تفاوت خواهد داشت. پرواضح است که همچین چیزی باعث سردرگمی ما و سایرین می‌شود، پس بهتر است برای جلوگیری از هرگونه مشکل، همه‌ی یوزرنیم‌ها را با حروف کوچک ایجاد کنیم.

در لینوکس، هر یوزرنیم علاوه بر نام، یک عدد منحصر به فرد نیز دارد که لینوکس با آن عدد یوزرنیم را شناسایی می‌کند. به این عدد User ID یا UID می‌گویند.

آشنایی با گروه‌ها و کاربرد آنها در لینوکس

لینوکس از گروه‌ها برای سازماندهی کاربران استفاده می‌کند. به طور کلی گروه‌ها بسیار شبیه به یوزرنیم‌ها می‌باشند؛ یعنی دقیقا مانند یوزرنیم‌ها تعریف می‌شوند، نام‌هایی شبیه به یوزرنیم‌ها دارند و همچنین مانند یوزرنیم‌ها با یک سری عدد شناسایی می‌شوند. هر گروه در لینوکس، می‌تواند از صفر تا بی‌نهایت عضو داشته باشد (بی‌نهایت یعنی به تعداد کلیه‌ی یوزرهای موجود در سیستم) و هر کاربر نیز می‌تواند عضو یک تا چند گروه باشد. هر کاربر، یک گروه پیش‌فرض (یا گروه اصلی) دارد و زمانی که کاربر یک فایل ایجاد می‌کند، آن فایل به گروه پیش‌فرض (یا گروه اصلی) کاربر ایجاد کننده‌ی فایل تعلق پیدا می‌کند.

گروه‌ها ما را در مدیریت دسترسی به فایل‌ها و دایرکتوری‌ها یاری می‌دهند و بدین ترتیب، در امنیت سیستم نقش دارند. اگر به خاطر داشته باشید، هر فایل یا دایرکتوری در لینوکس، به یک کاربر و یک گروه تعلق دارد و همچنین هر فایل و هر دایرکتوری، مجوزهایی به کاربر مالک و گروه مالک فایل می‌دهند.

ساختار گروه‌ها در لینوکس به ما اجازه می‌دهد که بتوانیم یک سیستم امنیتی مناسب را برای همکاری دسته‌ای از کاربران روی یک فایل، به وجود آوریم. همچنین این ساختار باعث می‌شود که بتوانیم دسته‌ای از کاربران را از دسترسی به یک سری فایل، منع کنیم. مثلا ما می‌توانیم برای پروژه‌های متفاوت، گروه‌های متفاوت ایجاد کنیم، به طوری که اعضای هر پروژه، عضو گروه مربوط به آن پروژه خواهند بود و به فایل‌های سایر پروژه‌ها دسترسی نخواهند داشت، یا مثلا می‌توانیم اجازه‌ی دسترسی به یک سخت‌افزار نظیر پرینتر را فقط به اعضای یک گروه خاص بدهیم (چرا که سخت‌افزارها در لینوکس به صورت یک فایل نمایش داده می‌شوند).

آشنایی با UID، GID و ارتباط آنها با یوزرنیم‌ها و گروه‌ها

همانطور که قبلا اشاره کردیم، لینوکس یوزرنیم‌ها و گروه‌ها را برای خود، با اعداد شناسایی و مدیریت می‌کند. به این اعداد شماره‌ی کاربر (User ID یا UID) و شماره‌ی گروه (Group ID یا GID) می‌گویند. برای مثال یوزرنیمی که ما به عنوان behnam می‌شناسیم، برای لینوکس با شماره‌ی UID آن (مثلا ۱۰۰۱) شناسایی می‌شود. معمولا، لینوکس به صورت اتوماتیک چگونگی اختصاص UID و GID به کاربران و گروه‌ها را مدیریت می‌کند و نیازی به مداخله‌ی ما ندارد؛ یعنی ما از یوزرنیم و نام گروه استفاده می‌کنیم و لینوکس آن نام‌ها را به شماره تبدیل می‌کند.

اکثر توزیع‌های لینوکسی، معمولا تعدادی از UID و GIDها را به صورت رزرو برای حساب‌های کاربری سرویس‌ها نگه می‌دارند. تعداد این UID و GIDهای رزرو از توزیع به توزع متفاوت می‌باشد و معمولا عددی فراتر از ۱۰۰ می‌باشد. مثلا CentOS تعداد ۱۰۰۰ UID و GID را به صورت رزرو برای حساب‌های کاربری سرویس‌ها رزرو دارد و کلیه‌ی یوزرنیم‌هایی که به صورت دستی ایجاد می‌کنیم، دارای شماره UID ۱۰۰۰ به بالا می‌باشند. مهم‌ترین UID و GID رزرو شده، ۰ می‌باشد. این شماره به یوزرنیم root و همچنین گروه root تعلق دارد و از قبل می‌دانیم که root یوزی است که می‌تواند هر کاری را در سیستم انجام دهد.

نکته: محدودیت‌های شماره‌گذاری UID و GID در فایل /etc/login.def تعریف می‌شوند. ما در بخش‌های بعدی با این فایل بیشتر آشنا می‌شویم.

ممکن است از خود بپرسید که اگر یک حساب‌کاربری را حذف کنیم، چه اتفاقی برای UID و GID آن میافتد؟ ما می‌توانیم از UID و GID حساب‌کاربری حذف شده استفاده کنیم و آنها را به یک حساب کاربری جدید بدهیم،

اما معمولا اکثر ابزارهای ایجاد حساب‌کاربری، از UID و GIDهای حساب‌های کاربری حذف شده استفاده نمی‌کنند. این باعث می‌شود که در شماره‌گذاری UID و GID، یک فاصله (یا gap) به وجود آید. این فاصله هیچ مشکلی برای ما ایجاد نمی‌کند، مگر این که به هر دلیلی، انقدر اکانت روی سیستم داشته باشیم که هیچ شماره‌ی جدیدی برای UID و GID باقی نمانده باشد. نکته‌ی جالب این است که استفاده از UID و GID اکانت‌های حذف شده می‌تواند برای ما مشکل‌ساز شود؛ چرا که خیلی از اوقات ممکن است حساب کاربری حذف شده، تعدادی فایل در سیستم داشته باشد و ما با ارائه‌ی UID آن کاربر به یک کاربر جدید، آن کاربر را تبدیل به مالک فایل کنیم و بدین ترتیب، مشکلات امنیتی متعددی برای سیستم خود به وجود آوریم.

از طرفی دیگر، ما می‌توانیم چندین یوزرنیم با UID یکسان و همچنین چندین گروه با GID یکسان داشته باشیم. این کاربران با هم از نظر یوزرنیم، رمز مورد اسفاده برای ورود به سیستم و حتی موقعیت دایرکتوری Home تفاوت خواهند داشت، اما از نظر مالکیت فایل‌ها، با هم یکسان خواهند بود. بهتر است از انجام چنین کاری، مگر در شراط اضطراری، پرهیز کنیم.

نکته: به طور کلی از کرنل نسخه‌ی ۲,٤ به بعد، ما می‌توانیم تا ٤,۲ میلیارد حساب کاربری در یک سیستم لینوکسی داشته باشیم.

بررسی فایل‌های درگیر در ایجاد حساب‌های کاربری جدید

یکی از مواردی که در بخش قبل به آن اشاره کردیم، این بود که لینوکس تعدادی از UIDها را برای حساب‌های کاربری سیستمی رزرو کرده است و در نتیجه‌ی آن، شماره‌ی UID حساب‌های کاربری معمولی از یک عدد خاص آغاز می‌شود. تا به حال فکر کرده‌اید که سیستم از کجا می‌فهمد که چه تعداد UID را به صورت رزور نگه دارد؟

از طرفی دیگر، همانطور که در بخش بعد خواهید دید، ما برای ایجاد حساب‌های کاربری جدید در سیستم، معمولا از تعدادی دستور متفاوت استفاده می‌کنیم. مثلا برای ایجاد یوزرنیم جدید، از دستوری به نام `useradd` استفاده می‌کنیم، یا برای قرار دادن رمز روی آن یوزرنیم، از دستور `passwd` استفاده کنیم. این دستورها از کجا میدانند که باید چه UID را به حساب کاربری اختصاص دهند یا در چه روزی رمز یک حساب کاربری را منقضی کنند؟

این دستورها برای تشخصی چگونگی عملکرد، به محتویات چندین فایل و دایرکتوری نگاه می‌کند و بر اساس اطلاعات موجود در آنها، اقدام به ایجاد یک حساب کاربری می‌کند. ما در این بخش می‌خواهیم با این فایل‌ها و دایرکتوری‌ها آشنا شویم.

فایل `/etc/login.defs`

فایل `/etc/login.defs` دستورالعمل‌های مورد نیاز برای ابزارهای ایجاد حساب‌های کاربری را درون خود دارد. این فایل تقریبا روی همه‌ی توزیع‌های لینوکسی به صورت پیش‌فرض موجود می‌باشد. دستورالعمل‌های موجود در این فایل، طول پسوورد، مدت اعتبار پسورد، ایجاد یا عدم ایجاد دایرکتوری Home برای کاربران جدید و... را کنترل می‌کنند. این دستورالعمل‌ها، توسط یک سری متغیر مشخص می‌شوند. برخی از این متغیرها به صورت پیش‌فرض فعال هستند و برخی غیرفعال می‌باشند. بیایید نگاهی به متغیرهای فعال این فایل بیاندازیم:

```
[root@localhost ~]# grep -v "^#" /etc/login.defs
MAIL_DIR                /var/spool/mail
PASS_MAX_DAYS           99999
PASS_MIN_DAYS           0
PASS_MIN_LEN            5
PASS_WARN_AGE           7
UID_MIN                 1000
UID_MAX                 60000
SYS_UID_MIN             201
SYS_UID_MAX             999
GID_MIN                 1000
GID_MAX                 60000
SYS_GID_MIN             201
SYS_GID_MAX             999
CREATE_HOME             yes
UMASK                   077
USERGROUPS_ENAB         yes
ENCRYPT_METHOD          SHA512
```

از این میان، مهم‌ترین متغیرها به شرح زیر می‌باشند:

جدول ۵- متغیرهای مهم در /etc/login.defs

متغیر	عملکرد
PASS_MAX_DAYS	مشخص می‌کند که یک رمز به مدت چند روز معتبر می‌باشد. پس از گذر مدت زمان مشخص شده توسط این متغیر، کاربر باید رمز خود را تغییر دهد.
PASS_MIN_DAYS	مشخص می‌کند که چند روز باید از آخرین تغییر رمز توسط کاربر بگذرد تا او بتواند بار دیگر رمز خود را تغییر دهد.
PASS_MIN_LENGTH	حداقل تعداد کاراکتری که باید در رمز موجود باشد را مشخص می‌کند.
PASS_WARN_AGE	مشخص می‌کند که چند روز مانده به روز پایان اعتبار رمز، هشدار تغییر رمز به کاربر داده می‌شود (هنگام لاگین به سیستم).
CREATE_HOME	اگر مقدار آن برابر با yes باشد، یک دایرکتوری Home برای هر حساب کاربری جدید ایجاد می‌کند.
ENCRYPT_METHOD	روش مورد استفاده برای Hash کردن رمزها را مشخص می‌کند.
UID_MIN	کمترین مقدار UID اختصاص یافته به یوزرنیم‌های معمولی (حساب‌های کاربری افراد) را مشخص می‌کند.
UID_MAX	بالاترین مقدار UID اختصاص یافته به یوزنیم‌های معمولی را مشخص می‌کند.
SYS_UID_MIN	کمترین مقدار UID اختصاص یافته به یوزرنیم‌های سیستمی را مشخص می‌کند.
SYS_UID_MAX	بالاترین مقدار UID اختصاص یافته به یوزرنیم‌های سیستمی را مشخص می‌کند.

فایل /etc/default/useradd

فایل /etc/default/useradd، یکی دیگر از فایل‌هایی است که در فرآیند ایجاد یوزرنیم‌های جدید دخیل می‌باشد. این فایل از /etc/login.defs بسیار کوتاه تر بوده و محتویات آن به شرح زیر می‌باشد:

```
[root@localhost ~]# cat /etc/default/useradd
# useradd defaults file
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=yes
```

شاید جالب‌ترین متغیر در این فایل، متغیر HOME باشد. این متغیر که اکنون مقدار /home را دارد، مشخص می‌کند که دایرکتوری Home هر کاربر، باید در چه دایرکتوری ایجاد شود. در جدول زیر، برخی از متغیرهای مهم در /etc/default/useradd و همچنین عملکرد آنها را مشاهده می‌کنیم:

جدول ۶- متغیرهای مهم در /etc/default/useradd

متغیر	عملکرد
HOME	موقعیت دایرکتوری پایه برای دایرکتوری Home حساب‌های کاربری را مشخص می‌کند.
INACTIVE	مشخص می‌کند که چند روز باید از اتمام اعتبار رمز کاربر و عدم تغییر آن توسط او گذشته باشد تا حساب کاربری او به صورت کامل، غیرفعال شود.
SKEL	موقعیت دایرکتوری Skeleton را مشخص می‌کند. در بخش بعد در مورد این مفهوم صحبت می‌کنیم.
SHELL	شِل پیش‌فرض کاربران را مشخص می‌کند (bash، fish، zsh و...). معمولا مقدار آن برابر با /bin/bash می‌باشد. ما قبلا عملکرد شِل را توضیح داده‌ایم، اما به طور کلی، شِل همان محیطی است که در آن دستورات متفاوت را وارد می‌کنیم.

دایرکتوری /etc/skel

دایرکتوری /etc/skel که به آن دایرکتوری Skeleton نیز می‌گویند، دایرکتوری است که می‌توانیم درون آن یک سری فایل قرار دهیم. هنگام ایجاد حساب‌های کاربری، کلیه‌ی فایل‌های موجود در این دایرکتوری، در دایرکتوری Home حساب‌های کاربری ایجاد شده کپی خواهد شد. بیایید نگاهی به محتوای این دایرکتوری در سیستم CentOS 7 Minimal بیاندازیم:

```
[root@localhost ~]# ls -la /etc/skel/
total 32
drwxr-xr-x.  2 root root   90 Apr 12  2020 .
drwxr-xr-x. 93 root root 8192 Oct 29 09:59 ..
-rw-r--r--.  1 root root   18 Aug  8  2019 .bash_logout
-rw-r--r--.  1 root root  193 Aug  8  2019 .bash_profile
-rw-r--r--.  1 root root  231 Aug  8  2019 .bashrc
```

همانطور که می‌بینید، ما با استفاده از آپشن la- (l به معنای نشان دادن خروجی صورت لیست و a به معنای نشان دادن فایل‌های پنهان)، فایل‌های موجود در دایرکتوری Skeleton این سیستم را مشاهده کردیم.

کلیه‌ی فایل‌های موجود در اینجا (که فایل‌های تنظیمات مربوط به bash می‌باشند) در دایرکتوری Home همه‌ی کاربران جدیدی که ایجاد می‌کنیم قرار خواهند گرفت.
توجه کنید که اگر فایل جدیدی را در /etc/skel قرار دهیم، فایل‌های جدید فقط در دایرکتوری Home حساب‌های کاربری که از الان به بعد ایجاد می‌کنیم قرار خواهند گرفت (یا به عبارت دیگر، این فایل‌های جدید در دایرکتوری Home کاربران کنونی سیستم قرار نخواهد گرفت).

نکته: محیط ترمینال و کنسول لینوکس، دقیقا مانند سایر سرویس‌های لینوکسی، با استفاده از یک سری فایل متنی تنظیم می‌شود. همانطور که می‌دانید، تا به اینجا برای کار با سیستم، از محیط شِل bash استفاده کرده‌ایم.
برای تنظیم bash، از فایل‌های زیر استفاده می‌کنیم:

- /etc/profile
- فایل‌های موجود در /etc/profile.d
- /etc/bash.bashrc
- /etc/bashrc
- ~/.bashrc
- ~/.bash_profile
- ~/.bash_login
- ~/.profile

فایل‌هایی که در دایرکتوری /etc قرار دارند، فایل‌های تنظیمات گلوبال می‌باشند، یعنی تنظیماتی هستند که روی محیط کاربری همه‌ی کاربران تاثیر خواهند داشت. فایل‌هایی که در دایرکتوری Home کاربران (~) قرار دارند، فقط روی محیط آن کاربر خاص تاثیر خواهند داشت و توسط خود کاربران نیز قابل تنظیم می‌باشند.
همه‌ی این فایل‌ها، بخش‌های متفاوت bash، یعنی مواردی نظیر رنگ صفحه، رنگ نوشته‌ها، و حتی موارد تخصصی‌تری نظیر متغیرهای محیطی (Environment Variables) را تنظیم می‌کنند. مدیر سیستم می‌تواند فایل‌های تنظیمات گلوبال را دست‌کاری کرده و متغیرهای محیطی که همه‌ی کاربران به آن دسترسی دارند را تغییر دهد یا مواردی به آن اضافه کند (ما قبلا در مورد متغیرهای محلی صحبت کرده‌ایم، پس دیگر به توضیح آنها نمی‌پردازیم).
گفتیم که کلیه‌ی فایل‌های موجود در دایرکتوری Skeleton، در دایرکتوری Home کلیه‌ی کاربران جدید قرار می‌گیرد، پس علاوه بر تغییر فایل‌های تنظیمات گلوبال، مدیر سیستم می‌تواند فایل‌های تنظیمات موجود در دایرکتوری Skeleton را تغییر داده و بدین ترتیب، محیط ایجاد شده برای کاربران جدید را بدون دستکاری تنظیمات گلوبال، تغییر دهد.

حال که با فایل‌ها و دایرکتوری‌هایی که هنگام ایجاد حساب‌های کاربری جدید از آنها استفاده می‌شود آشنا شدیم، نوبت آن رسیده که در مورد فایل‌هایی که هنگام ایجاد حساب‌های کاربری جدید دچار تغییر می‌شوند صحبت کنیم. این فایل‌ها، مانند یک دیتابیس عمل می‌کنند و کلیه‌ی اطلاعات مربوط به حساب‌های کاربری متفاوت را درون خود ذخیره می‌کنند. در بخش‌های بعد، با این فایل‌ها آشنا شده و آنها را به صورت کامل بررسی می‌کنیم و علاوه بر آن، در مورد چگونگی جستجو در این فایل‌ها صحبت خواهیم کرد.

فایل /etc/passwd

فایل /etc/passwd اطلاعات مربوط به همه‌ی حساب‌های کاربری را درون خود دارد. در این فایل، اطلاعات هر حساب کاربری، یک خط مجزا را اشغال می‌کنند. در واقع زمانی که ما یک حساب کاربری جدید ایجاد می‌کنیم، یک خط جدید که شامل اطلاعات مربوط به این حساب کاربری می‌باشد به این فایل اضافه می‌شود. بیایید نگاهی به محتویات این فایل بیاندازیم:

```
[root@localhost ~]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
...
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
postfix:x:89:89::/var/spool/postfix:/sbin/nologin
```

هر خط موجود در این فایل، از هفت فیلد تشکیل شده است. این فیلدها، با علامت دونقطه (:) از هم جدا شده‌اند. اطلاعاتی که در هر فیلد می‌گیرد به شرح زیر می‌باشد:

جدول ۷- توصیف فیلدهای موجود در /etc/passwd

شماره فیلد	توصیف
۱	یوزرنیم حساب‌کاربری را نشان می‌دهند.
۲	رمز اختصاص یافته به یک یوزرنیم را نشان می‌دهد. امروزه از این فایل برای ذخیره‌ی رمزها استفاده نمی‌شود و به جای رمز، حرف x در این فیلد مشاهده می‌شود. وجود حرف x در این فیلد، یعنی رمزها در /etc/shadow ذخیره شده‌اند.
۳	UID حساب کاربری را نشان می‌دهد.
٤	GID حساب کاربری را نشان می‌دهد.
۵	این فیلد برای کامنت می‌باشد. معمولا در این قسمت، نام کامل کاربر قرار می‌گیرد.
۶	موقعیت دایرکتوری Home کاربر را نشان می‌دهد.
۷	شِل پیش‌فرض کاربر را مشخص می‌کند. اگر مقدار موجود در این فیلد برابر با /sbin/nologin یا /bin/false باشد، کاربر نمی‌تواند به سیستم لاگین کند.

اگر به فیلد هفتم در فایل /etc/passwd سیستم خود نگاه کنید، می‌بینید که بسیاری از خطوط موجود در این فایل، شِل پیش‌فرض /sbin/nologin یا /bin/false را دارند. وجود این شِل برای یک حساب کاربری، یعنی آن کاربر نمی‌تواند به سیستم وارد شود. /sbin/nologin معمولا به عنوان شِل حساب‌های کاربری مربوط به سرویس‌های سیستمی انتخاب می‌شود. سرویس‌های سیستمی به حساب کاربری نیاز دارند، اما این حساب‌های کاربری نیازی به لاگین کردن درون سیستم ندارند.

نکته: اگر فردی بتواند با یکی از این حساب‌هایی که شِل /sbin/nologin دارند درون سیستم لاگین کند، سیستم به سرعت آن کاربر را Log Out می‌کند و پیغامی به او نشان می‌دهد. در برخی از توزیع‌ها، می‌توانیم این پیغام را تغییر دهیم. این کار را با ایجاد فایلی به نام /etc/nologin.txt انجام می‌دهیم. انتخاب

/bin/false به عنوان شِل یک حساب کاربری، باعث می‌شود که سیستم کمی خشن‌تر عمل کند و کاربر را بدون نشان دادن هر گونه پیغامی، Log Out کند.

به طور خلاصه، می‌توان گفت که مهم‌ترین نکته در مورد /etc/passwd این است که به دلیل Plain Text بودن این فایل، رمزها درون آن ذخیره نمی‌شوند، بلکه رمزها در فایل امن‌تر /etc/shadow ذخیره می‌شوند.

فایل /etc/shadow

فایل /etc/shadow یکی دیگر از فایل‌هایی است که هنگام ایجاد حساب‌های کاربری آپدیت می‌شود. این فایل، اطلاعاتی در مورد رمز هر حساب کاربری را درون خود دارد (حتی اگر رمزی روی آن حساب کاربری قرار نداده باشیم). در این فایل نیز، هر خط، اطلاعات مربوط به یک حساب کاربری را درون خود دارد. به طور کلی، محتوای این فایل به صورت زیر می‌باشد:

```
[root@localhost ~]# cat /etc/shadow
root:$6$50[...]::0:99999:7:::
bin:*:17834:0:99999:7:::
daemon:*:17834:0:99999:7:::
...
sshd:!!:18341::::::
postfix:!!:18341::::::
```

همانطور که می‌بینید، این فایل نیز از تعدادی فیلد تشکیل شده و هر فیلد، با یک علامت دونقطه (:) از فیلد قبلی جدا شده است. به طور کلی، در هر خط، ۹ فیلد وجود دارد که هر کدام به شرح زیر می‌باشند:

جدول ۸- توصیف فیلدهای موجود در /etc/shadow

شماره فیلد	توصیف
۱	یوزرنیم یک حساب‌کاربری را نشان می‌دهد.
۲	رمز اختصاص یافته به حساب کاربری را به صورت Salt و Hash نشان می‌دهد. علامت !! یا ! در این فیلد، یعنی هیچ رمزی روی این اکانت قرار نگرفته است. علامت ! یا * در این فیلد، یعنی این حساب کاربری نمی‌تواند با استفاده از یک رمز وارد سیستم شود. وجود یک علامت ! در ابتدای رمز، یعنی این حساب کاربری قفل شده است.
۳	تاریخ آخرین باری که رمز این حساب کاربری تغییر یافته را مشخص می‌کند (در فرمت Unix Time).
٤	تعداد روزهایی که باید از تغییر رمز بگذرد تا کاربر بتواند بار دیگر رمز را عوض کند را مشخص می‌کند.
۵	تعداد روزهایی که باید بگذرد تا رمز کنونی منقضی شود را مشخص می‌کند. این، در واقع تاریخ انقضای رمز می‌باشد.
۶	مشخص می‌کند که چند روز مانده به روز پایان اعتبار رمز، هشدار تغییر رمز به کاربر داده می‌شود.

۷	تعداد روزهایی که از منقضی شدن رمز و عدم تغییر آن توسط کاربر باید بگذرد تا حساب‌کاربری او به طور کامل غیرفعال شود را مشخص می‌کند.
۸	تاریخ انقضای حساب کاربری را در فرمت Unix Time (به روز) مشخص می‌کند.
۹	به این فیلد Special Flag می‌گویند. این فیلد برای استفاده‌ی خاص رزرو شده است. در حال حاضر از این فیلد استفاده نمی‌شود و معمولا خالی می‌باشد.

نکته: Unix Time یا Unix Epoch Time (برخی به آن POSIX Time هم می‌گویند) تعداد ثانیه‌هایی است که از روز اول ژانویه‌ی ۱۹۷۰ تا کنون سپری شده است. در فایل /etc/shadow، این زمان به جای این که در واحد ثانیه باشد، در واحد روز می‌باشد. سالیان زیادی است که از Unix Time در سیستم‌های یونیکسی و لینوکسی استفاده می‌کنند و به احتمال خیلی زیاد در سال ۲۰۳۸، استفاده از Unix Time منجر به مشکلات اساسی نظیر مشکل معروف سال ۲۰۰۰ می‌شود. برای اطلاعات بیشتر می‌توانید به این لینک مراجعه کنید. برای تبدیل Unix Time به زمان قابل درک توسط انسان، کافی است از دستور chage استفاده کنیم.

مهم است که تفاوت بین انقضای رمز و انقضای حساب کاربری (یا غیر فعال شدن حساب کاربری) را درک کنیم. زمانی که رمز یک حساب کاربری منقضی می‌شود، معمولا به کاربر ضرب‌الاجلی داده می‌شود که با رمز قدیمی وارد حساب کاربری خود شده و رمز خود را تغییر دهد. اگر کاربری در طی این ضرب‌الاجل رمز خود را تغییر ندهد، حسابش قفل می‌شود.

زمانی که حساب کاربری یک کاربر منقضی شود، هیچ ضرب‌الاجلی وجود ندارد و پس از منقضی شدن، کاربر نمی‌تواند وارد حساب کاربری خود شود.

نکته: زمانی که یک سری کاربر قرار است به صورت موقت از سیستم استفاده کنند، مشخص کردن یک زمان انقضا برای حساب‌های کاربری بسیار کارآمد می‌باشد. در این حالت، اگر فراموش کنیم که حساب کاربری یک کاربر موقت را حذف کنیم، آن حساب به صورت اتوماتیک غیرفعال شده و باعث به وجود آمدن مشکلات امنیتی نمی‌شود.

فایل /etc/group

اطلاعات مربوط به کلیه‌ی گروه‌ها در سیستم، درون فایل /etc/group قرار دارد. این فایل بسیار شبیه به فایل‌های /etc/passwd و /etc/shadow می‌باشد. بیایید نگاهی به محتویات این فایل بیاندازیم:

```
[root@localhost ~]# cat /etc/group
root:x:0:
bin:x:1:
daemon:x:2:
sys:x:3:
...
postdrop:x:90:
postfix:x:89:
```

در این فایل، اطلاعات مربوط به هر گروه، در یک خط قرار می‌گیرد و هر خط، شامل ٤ فیلد می‌باشد. به طوری که:

- فیلد اول نشان دهنده‌ی نام گروه می‌باشد.
- فیلد دوم نشان دهنده‌ی پسوورد گروه می‌باشد. وجود حرف x در این فیلد، یعنی رمز این گروه در فایل /etc/gshadow ذخیره شده است.
- فیلد سوم نشان دهنده‌ی GID این گروه می‌باشد.
- فیلد چهارم، یوزرنیم کلیه‌ی کاربرانی که عضو این گروه هستند را نشان می‌دهد.

فایل /etc/gshadow

این فایل، اطلاعاتی در مورد رمز گروه‌ها و همچنین مدیران یک گروه را درون خود دارد. این فایل نیز بسیار شبیه به /etc/passwd و /etc/shadow می‌باشد. بیایید نگاهی به محتویات این فایل بیاندازیم:

```
[root@localhost ~]# cat /etc/gshadow
root:::
bin:::
daemon:::
sys:::
...
postdrop:!::
postfix:!::
```

همانطور که می‌بینید، این فایل نیز از ٤ فیلد تشکیل شده است. به طوری که:

- فیلد اول نشان دهنده‌ی نام گروه می‌باشد.
- فیلد دوم نشان دهنده‌ی رمز گروه می‌باشد. علامت ! در این فیلد به معنای عدم وجود رمز برای این گروه می‌باشد.
- فیلد سوم نشان دهنده‌ی یوزرنیم مدیران این گروه می‌باشد. مدیران گروه می‌توانند اعضای گروه و همچنین رمز گروه را تغییر دهند. هر گروه می‌تواند بیش از یک مدیر داشته باشد. یوزرنیم‌های مدیران گروه در این فیلد، با علامت کاما (,) از هم جدا می‌شود.
- فیلد چهارم نشان دهنده‌ی یوزرنیم اعضای این گروه می‌باشد.

ایجاد و مدیریت حساب‌های کاربری و گروه‌ها

ما تا به اینجا با مفاهیم کلی مربوط به حساب‌های کاربری و گروه‌ها آشنا شدیم. حال نوبت آن رسیده که در مورد چگونگی ایجاد یوزرنیم‌ها و گروه‌ها صحبت کنیم. در این بخش، با ابزارهایی که برای ایجاد و مدیریت حساب‌های کاربری وجود دارند، آشنا خواهیم شد.

ایجاد، تغییر و مدیریت یوزرنیم و پسوورد

همانطور که گفتیم، برای ایجاد و همچنین مدیریت حساب کاربری، از چندین دستور متفاوت استفاده می‌کنیم. در این قسمت می‌خواهیم به شرح کامل این دستورها و نحوه‌ی عملکرد آنها بپردازیم.

ایجاد یوزرنیم‌های جدید با *useradd*

ایجاد یوزرنیم در سیستم‌های لینوکسی، با دستور useradd صورت می‌پذیرد. بیایید بدون هیچ اتلاف وقت، از این دستور استفاده کرده و یک یوزرنیم جدید به نام thealbatross ایجاد کنیم:

```
[root@localhost ~]# useradd thealbatross
```

همانطور که می‌بینید، اضافه کردن یک کاربر جدید با استفاده از useradd، به همین سادگی می‌باشد. بیایید با نگاه کردن به فایل /etc/passwd از ایجاد این یوزرنیم اطمینان حاصل کنیم:

```
[root@localhost ~]# grep "^thealbatross" /etc/passwd
thealbatross:x:1000:1000::/home/thealbatross:/bin/bash
```

این امر را می‌توانیم با نگاه کردن به فایل /etc/shadow نیز بررسی کنیم:

```
[root@localhost ~]# grep "^thealbatross" /etc/shadow
thealbatross:!!:18567:0:99999:7:::
```

اگر خروجی دستور بالا را نگاه کنیم، می‌بینیم که در فیلد رمز (فیلد دوم)، علامت !! قرار گرفته است. همانطور که در جدول ۸ دیدیم، این علامت بدین معناست که رمزی روی این حساب کاربری قرار نگرفته است. ما در بخش‌های بعدی در مورد چگونگی قرار دادن رمز روی یک حساب کاربری صحبت می‌کنیم.

بیایید به دایرکتوری Home کاربری که ایجاد کردیم نگاهی بیاندازیم:

```
[root@localhost ~]# ls -la /home/thealbatross/
total 20
drwx------. 2 thealbatross thealbatross  90 Nov  1 11:22 .
drwxr-xr-x. 4 root         root          82 Nov  1 11:22 ..
-rw-r--r--. 1 thealbatross thealbatross  18 Aug  8  2019 .bash_logout
-rw-r--r--. 1 thealbatross thealbatross 193 Aug  8  2019 .bash_profile
-rw-r--r--. 1 thealbatross thealbatross 231 Aug  8  2019 .bashrc
```

```
[root@localhost ~]# ls -la /etc/skel/
total 32
drwxr-xr-x.  2 root root   90 Apr 12  2020 .
drwxr-xr-x. 93 root root 8192 Nov  1 11:22 ..
-rw-r--r--.  1 root root   18 Aug  8  2019 .bash_logout
-rw-r--r--.  1 root root  193 Aug  8  2019 .bash_profile
-rw-r--r--.  1 root root  231 Aug  8  2019 .bashrc
```

همانطور که می‌بینید، فایل‌های موجود در دایرکتوری حساب کاربری که ایجاد کردیم، دقیقا همان فایل‌هایی هستند که در دایرکتوری /etc/skel قرار دارند. این دقیقا همان مسئله‌ای است که در بخش‌های قبل و هنگام معرفی /etc/skel به آن اشاره کردیم.

همانطور که دیدیم، انجام‌کارهای ساده با دستور useradd بسیار ساده می‌باشد، اما این دستور قابلیت‌های پیشرفته‌تری هم دارد که برخی از آنها را در جدول زیر مشاهده می‌کنیم:

جدول ۹- برخی از آپشن‌های دستور useradd

کوتاه	بلند	عملکرد
-c	**--comment**	محتویات موجود در فیلد کامنت (فیلد پنجم) در /etc/passwd را مشخص می‌کند. معمولا شامل نام کامل کاربر می‌باشد.
-d	**--home یا --homedir**	با استفاده از آن، دایرکتوری Home کاربر را مشخص می‌کنیم. در صورت عدم استفاده از این آپشن، useradd به دستورالعمل‌های CREATE_HOME و HOME که در فایل‌های /etc/login.defs و /etc/default/useradd قرار دارند، نگاه می‌کند..

-D	**--defaults**	دستورالعمل‌های موجود در /etc/default/useradd را به ما نشان می‌دهد.
-e	**--expiredate**	تاریخ انقضای حساب کاربری را در فرمت YYYY-MM-DD مشخص می‌کند. در صورت عدم استفاده از این آپشن، به محتویات دستورالعمل EXPIRE در فایل /etc/default/useradd نگاه می‌کند.
-f	**--inactive**	تعداد روزهایی که از اتمام اعتبار رمز و عدم تغییر آن توسط کاربر باید بگذرد تا حساب کاربری غیر فعال شود را مشخص می‌کند. ارائه‌ی عدد ۱- به این آپشن، به معنای این می‌باشد که حساب کاربری هیچ وقت نباید غیرفعال شود. در صورت عدم استفاده از این آپشن، از محتویات دستورالعمل INACTIVE که در فایل /etc/default/useradd قرار دارد، استفاده می‌شود.
-g	**--gid**	گروه اصلی (پیش‌فرض) حساب کاربری را مشخص می‌کند.
-G	**--groups**	سایر گروه‌هایی که حساب کاربری باید در آنها عضو شود را مشخص می‌کند.
-m	**--create-home**	برای کاربر یک دایرکتوری Home ایجاد می‌کند. در صورت عدم استفاده از این آپشن، طبق محتویات دستورالعمل CREATE_HOME در فایل /etc/login.defs عمل می‌شود.
-M	**--no-create-home**	برای کاربر دایرکتوری Home ایجاد نمی‌کند. در صورت عدم استفاده از این آپشن، طبق محتویات دستورالعمل CREATE_HOME در فایل /etc/login.defs عمل می‌شود.
-s	**--shell**	شِل حساب کاربری را مشخص می‌کند. در صورت عدم استفاده از این آپشن، شِل مشخص شده دردستورالعمل SHELL که در فایل /etc/default/useradd قرار دارد به کار برده می‌شود.
-u	**--uid**	UID حساب کاربری را مشخص می‌کند.
-r	**--system**	به جای حساب کاربری معمولی، حساب کاربری سیستمی ایجاد می‌کند.

بیایید از چندی از آپشن‌های useradd برای ایجاد یوزرنیم جدید استفاده کنیم. مثلا فرض کنید می‌خواهیم یک حساب کاربری جدید به نام puppy ایجاد کنیم، به طوری که این حساب کاربری دارای شِل zsh باشد، دارای UID برابر با ۱۲۳٤ باشد و در قسمت کامنت آن نیز، نام کامل او، یعنی Puppy Seeder نوشته شده باشد. برای این کار:

```
[root@localhost ~]# useradd -s /bin/zsh -u 1234 -c "Puppy Seeder" puppy
[root@localhost ~]# grep "puppy" /etc/passwd
puppy:x:1234:1234:Puppy Seeder:/home/puppy:/bin/zsh
```

همانطور که می‌بینید ما با استفاده از آپشن s-، شِل این کاربر را مشخص کردیم (موقعیت قرار گیری فایل باینری شِل). سپس با استفاده از آپشن u- به این دستور گفتیم که می‌خواهیم این کاربر، UID برابر با ۱۲۳٤ داشته باشد و در نهایت با آپشن c-، به این دستور گفتیم که کامنتی که برای این کاربر قرار می‌دهیم، برابر با Puppy Seeder می‌باشد. دقت کنید که چون بین کلمات به کار رفته در کامنت خود فاصله‌ی خالی (Space) داشتیم، کامنت را بین دو علامت "" قرار دادیم (حتی اگر فاصله‌ای بین کلمات موجود در کامنت وجود نداشته باشد، بهتر است آن را بین این دو علامت قرار دهیم). پس از مشخص کردن آپشن‌ها، یوزرنیم این حساب کاربری، یعنی puppy را مشخص و دستور را وارد کردیم. دقت کنید که هنگام استفاده از دستور useradd، بهتر است آپشن‌ها را پشت هم قرار ندهیم (یعنی ننویسیم suc-)، بلکه هر کدام را به صورت جداگانه وارد کرده، مقدار مورد نظر را به آن بدهیم و سپس به سراغ آپشن بعدی برویم.

استفاده از دستور *getent* برای جستجو در فایل‌های مربوط به حساب کاربری

قبل از ادامه‌ی صحبت در مورد حساب‌های کاربری، بهتر است در مورد دستور getent صحبت کنیم. برای جستجو در فایل‌هایی نظیر /etc/passwd و /etc/shadow مجبور به استفاده از دستور grep نیستیم. دستور getent، می‌تواند عمل جستجو در این دو فایل را بسیار ساده کند. برای استفاده از این دستور، کافی است پس از وارد کردن دستور، فقط نام فایل مورد نظر (مثلا passwd یا shadow) را وارد کرده و سپس نام حساب کاربری که می‌خواهیم اطلاعات آن را مشاهده کنیم را وارد کنیم. برای مثال:

```
[root@localhost ~]# getent passwd puppy
puppy:x:1234:1234:Puppy Seeder:/home/puppy:/bin/zsh
[root@localhost ~]# getent shadow puppy
puppy:!!:18572:10:60:20:3:18936:
```

همانطور که می‌بینید، ما ابتدا دستور getent را وارد کردیم، سپس نام فایلی که می‌خواهیم در آن جستجو کنیم را وارد کردیم و پس از آن، نام حساب کاربری مورد جستجو را وارد کردیم. پرواضح است که استفاده از getent بسیار ساده‌تر از استفاده از grep می‌باشد.

ایجاد و مدیریت پسوردها با *passwd*

زمانی که یک یوزرنیم جدید ایجاد می‌کنیم، باید بلافاصله یک پسورد برای آن یوزرنیم نیز ایجاد کنیم. ما این کار را با استفاده از دستور passwd انجام می‌دهیم. برای مثال، بیایید برای یوزرنیم puppy که در بخش قبل آن را ایجاد کردیم، یک پسورد ایجاد کنیم:

```
[root@localhost ~]# passwd puppy
Changing password for user puppy.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

همانطور که می‌بینید، با وارد کردن دستور passwd و سپس یوزرنیم مورد نظر، توانستیم روی آن یوزرنیم یک رمز قرار دهیم. توجه کنید که در صورت انتخاب رمزهای ساده، دستور passwd به شما هشداری مبنی بر ساده بودن رمز می‌دهد.

علاوه بر ایجاد رمز برای یوزرنیم‌های جدید، دستور passwd می‌تواند پسورد یوزرنیم‌هایی که از قبل در سیستم بوده‌اند را نیز تغییر دهد. برای این کار کافی است یوزرنیم مورد نظر را به passwd دهیم و پس از

وارد کردن رمز قبلی آن، رمز جدید را برای آن انتخاب کنیم. اگر بخواهیم پسوود یوزرنیم خودمان تغییر کند، کافی است دستور passwd را به تنهایی و بدون ارائه‌ی هیچ یوزرنیمی اجرا کنیم.

دستور passwd قابلیت‌هایی فراتر از ایجاد و تغییر پسوورد حساب‌های کاربری دارد. ما با استفاده از این دستور می‌توانیم یک حساب کاربری را قفل کنیم، برای یک حساب کاربری تاریخ انقضا قرار دهیم، پسوورد یک حساب کاربری را حذف کنیم و... . پرکاربردترین آپشن‌های passwd در جدول زیر قابل مشاهده می‌باشند:

جدول ۱۰- پرکاربردترین آپشن‌های passwd

کوتاه	بلند	عملکرد
-d	**--delete**	رمز حساب کاربری مشخص شده را حذف می‌کند.
-e	**--expire**	رمز حساب کاربری مشخص شده را منقضی می‌کند. در نتیجه‌ی آن، کاربر مشخص شده باید در اولین ورود خود به سیستم، رمز خود را تغییر دهد.
-i	**--inactive**	تعداد روزهایی که از اتمام اعتبار رمز و عدم تغییر آن توسط کاربر باید بگذرد تا حساب کاربری مشخص شده غیر فعال شود را مشخص می‌کند.
-l	**--lock**	این آپشن، یک علامت تعجب (!) در جلوی فیلد رمز حساب کاربری در فایل /etc/shadow قرار می‌دهد و حساب کاربری را قفل می‌کند. بدین ترتیب، کاربر نمی‌تواند وارد سیستم شود.
-n	**--minimum**	مشخص می‌کند که پسوورد، چند روز پس از تغییر می‌تواند دوباره تغییر کند.
-S	**--status**	وضعیت پسوورد حساب کاربری را نشان می‌دهد.
-u	**--unlock**	علامت تعجب (!) را از جلوی فیلد رمز حساب کاربری در فایل /etc/shadow بر میدارد و بدین ترتیب، آن حساب کاربری را از حالت قفل بودن خارج می‌کند.
-w	**--warning**	مشخص می‌کند که چند روز قبل از انقضای رمز یک حساب کاربری، باید به آن کاربر هشدار تغییر رمز (به صورت روزانه) نمایش داده شود.
-x	**--maximum**	مشخص می‌کند که چند روز پس از تغییر پسوورد یک حساب کاربری، آن پسورد منقضی می‌شود. به عبارت دیگر، زمان انقضای پسوورد را در واحد روز مشخص می‌کند.

از میان آپشن‌های موجود در جدول ۱۱، آپشن S- نیاز به توضیح بیشتری دارد. بیایید نمونه‌ای از خروجی این آپشن مشاهده کنیم:

```
[root@localhost ~]# passwd -S puppy
puppy PS 2020-11-04 0 99999 7 -1 (Password set, SHA512 crypt.)
```

همانطور که می‌بینید، در خروجی این دستور، وضعیت پسوورد حساب کاربری puppy آمده است. خروجی این دستور نیز به ۷ فیلد تقسیم شده است:

- فیلد اول، یوزرنیم را مشخص می‌کند.
- فیلد دوم، وضعیت پسوورد را مشخص می‌کند. این فیلد می‌تواند سه مقدار داشته باشد:
  - PS به معنای وجود پسوورد
  - NP به معنای عدم وجود پسوورد
  - LK به معنای پسوورد قفل شده
- فیلد سوم، نشان می‌دهد که آخرین بار، رمز در چه تاریخی تغییر داده شده است.
- فیلد چهارم نشان دهنده‌ی این است که رمز چند روز پس از تغییر می‌تواند دوباره تغییر کند.
- فیلد پنجم نشان می‌دهد که پسوورد پس از چند روز منقضی می‌شود.
- فیلد ششم مشخص می‌کند که از چند روز مانده به انقضای رمز، کاربر هشدار تغییر رمز را دریافت می‌کند
- فیلد هفتم، مشخص می‌کند که چند روز پس از منقضی شدن رمز و عدم تغییر آن توسط کاربر، این حساب کاربری غیرفعال می‌شود (مقدار ۱- یعنی این حساب کاربری هیچ وقت غیر فعال نمی‌شود).

مدیریت پسووردها با *chage*

همانطور که در بخش قبل دیدیم، دستور passwd آپشن‌های بسیار زیادی دارد و یادگیری همه‌ی آنها کار دشواری می‌باشد. ما می‌توانیم با استفاده از دستور chage، بسیاری از ویژگی‌های رمز یک حساب کاربری را تغییر دهیم. برای این کار، کافی است دستور chage را به علاوه‌ی یوزرنیمی که می‌خواهیم تنظیمات پسووردش را تغییر دهیم، وارد کنیم:

```
[root@localhost ~]# chage puppy
Changing the aging information for puppy
Enter the new value, or press ENTER for the default

        Minimum Password Age [0]: 10
        Maximum Password Age [99999]: 60
        Last Password Change (YYYY-MM-DD) [2020-11-04]: 2020-11-05
        Password Expiration Warning [7]: 20
        Password Inactive [-1]: 3
        Account Expiration Date (YYYY-MM-DD) [-1]: 2021-11-05
```

همانطور که می‌بینید، با استفاده chage توانستیم به صورت Interactive بیشتر تنظیماتی که باید با استفاده از آپشن‌های passwd به وجود می‌آوردیم را به صورت ساده‌تر روی یک حساب کاربری، اعمال کنیم.

دستور chage نیز آپشن‌های بسیاری دارد، اما شاید جالب‌ترین آپشن آن، آپشن l- باشد. این آپشن باعث می‌شود که دستور chage کلیه‌ی تنظیمات و مشخصات رمز یک حساب کاربری را به ما نشان دهد. برای استفاده، کافی است دستور chage را به همراه آپشن l- و نام یوزرنیمی که می‌خواهیم اطلاعات پسووردش را مشاهده کنیم را وارد کنیم:

```
[root@localhost ~]# chage -l puppy
Last password change                                    : Nov 04, 2020
Password expires                                        : never
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 7
```

همانطور که می‌بینید، این دستور، دقیقا همان اطلاعاتی که آپشن S- دستور passwd به ما نشان میداد را به ما ارائه کرد؛ با این تفاوت که خروجی این دستور بسیار خواناتر می‌باشد.

تغییر ویژگی‌های یک حساب کاربری با *usermod*

اگر بخواهیم ویژگی‌های یک حساب کاربری، مثل تاریخ انقضای آن و... را تغییر دهیم، از دستور usermod استفاده می‌کنیم. این دستور هم آپشن‌های بسیار زیادی دارد که در جدول زیر، پرکاربردترین آنها را مشاهده می‌کنید:

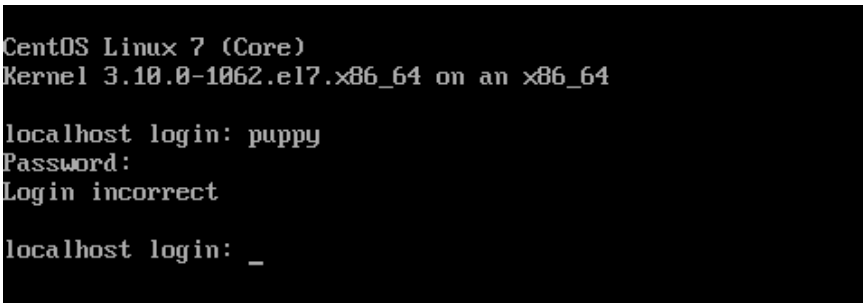
جدول ۱۱- پرکاربردترین آپشن‌های دستور usermod

کوتاه	بلند	عملکرد
-c	--comment	محتویات فیلد کامنت در /etc/passwd را تغییر می‌دهد. در این فیلد معمولا نام کامل کاربر ذخیره می‌شود.
-d	--home	یک دایرکتوری Home جدید برای کاربر مشخص می‌کند. ما می‌توانیم با استفاده از آپشن m- در کنار این آپشن، محتویات دایرکتوی Home کنونی کاربر را به دایرکتوری Home جدید او منتقل کنیم.
-e	--expiredate	تاریخ انقضای حساب کاربری را تغییر می‌دهد. تاریخ مورد نظر باید در فرمت YYYY-MM-DD نوشته شود.
-f	--inactive	تعداد روزهایی که از اتمام اعتبار رمز و عدم تغییر آن توسط کاربر باید بگذرد تا حساب کاربری غیر فعال شود را مشخص می‌کند. اگر به این آپشن مقدار ۱- را بدهیم، حساب کاربری هیچ وقت غیرفعال نمی‌شود.
-s	--shell	شِل حساب کاربری را تغییر می‌دهد.
-u	--uid	شماره‌ی UID حساب کاربری را تغییر می‌دهد.
-L	--lock	حساب کاربری را با قرار دادن یک علامت تعجب (!) جلوی رمز حساب کاربری در فایل /etc/shadow، قفل می‌کند.
-U	--unlock	حساب کاربری را با برداشتن علامت تعجب (!) موجود جلوی رمز آن در فایل /etc/shadow، از حالت قفل خارج می‌کند.
-l	--login	یوزرنیم حساب کاربری را تغییر می‌دهد، اما به موقعیت دایرکتوری Home آن حساب کاربری دست نمی‌زند. به عبارتی دیگر، با این که یوزرنیم عوض می‌شود، اما دایرکتوری Home همان دایرکتوری قبلی باقی خواهد ماند.
-g	--gid	گروه پیش‌فرض یک حساب کاربری را تغییر می‌دهد.
-G	--groups	کاربر را در گروه مشخص شده عضو می‌کند. استفاده از این آپشن به تنهایی، باعث می‌شود که کاربر از سایر گروه‌هایی که در آن عضو است خارج شده و فقط به عضویت گروه مشخص شده درآید. اگر بخواهیم کاربر عضو گروه‌های قبلی خود باقی بماند و عضو گروه مشخص شده نیز شود، به همراه این آپشن، از آپشن a- نیز استفاده می‌کنیم.

بیایید با استفاده از دستور usermod، حساب کاربری puppy را قفل کنیم:

```
[root@localhost ~]# usermod -L puppy
[root@localhost ~]# grep "puppy" /etc/shadow
puppy:!$6$japxID3D$esnNt3jxUMvIOLIDGd9Nz0:18572:10:60:20:3:18936:
```

همانطور که میبینید، در جلوی رمز این کاربر در فیلد پسوورد، یک علامت ! قرار گرفته است. حال اگر سعی کنیم با حساب کاربری puppy وارد سیستم شویم، با پیام Login Incorrect یا Authentication Failure مواجه خواهیم شد:



تصویر ۳- سعی برای ورود به حساب کاربری پس از قفل آن

برای این که حساب کاربری puppy را از حالت قفل خارج کنیم، از آپشن U- استفاده می‌کنیم:

```
[root@localhost ~]# usermod -U puppy
[root@localhost ~]# grep "puppy" /etc/shadow
puppy:$6$japxID3D$esnNt3jxUMvIOLIDGd9Nz0:18572:10:60:20:3:18936:
```

همانطور که می‌بینید، پس از وارد کردن این دستور، دیگر در جلو فیلد پسوورد در فایل /etc/shadow، علامت ! وجود ندارد و ما می‌توانیم به سادگی و مانند قبل وارد سیستم شویم.

حذف یک حساب کاربری با استفاده از *userdel*

ما می‌توانیم با استفاده از دستور userdel یک حساب کاربری را حذف کنیم. این دستور نیز آپشن‌های متعددی دارد، اما کاربردی‌ترین آنها، آپشن r- می‌باشد. استفاده از این آپشن باعث می‌شود که علاوه بر پاک کردن حساب کاربری، دایرکتوری Home و کلیه‌ی فایل‌ها و دایرکتوری‌های درون آن به طور کامل از روی سیستم پاک شوند. بیایید حساب کاربری thealbatross که آن را در بخش‌های قبل ایجاد کردیم را پاک کنیم. ابتدا بیایید مشخصات این حساب کاربری را به خاطر آوریم:

```
[root@localhost ~]# ls -a /home/thealbatross/
.  ..  .bash_logout  .bash_profile  .bashrc  .emacs  .zshrc
[root@localhost ~]# grep "thealbatross" /etc/passwd
thealbatross:x:1001:1001::/home/thealbatross:/bin/bash
```

حال که مشخصات این حساب کاربری را به خاطر آوردیم، با استفاده از دستور userdel و به کارگیری آپشن r-، اقدام به پاک کردن این حساب کاربری می‌کنیم:

```
[root@localhost ~]# userdel -r thealbatross
```

بیایید از صحت پاک شدن این حساب کاربری مطمئن شویم:

```
[root@localhost ~]# ls -a /home/thealbatross
ls: cannot access /home/thealbatross: No such file or directory
[root@localhost ~]# grep "thealbatross" /etc/passwd
```

همانطور که می‌بینید، پس از اجرای این دستور، هم دایرکتوری Home حساب کاربری thealbatross

پاک شده و همچنین مشخصات این حساب کاربری، دیگر در فایل /etc/passwd وجود ندارد.

مدیریت گروه‌ها

همانطور که قبلا گفتیم، گروه‌ها نیز دقیقا مثل یوزرنیم‌ها، هم با نام و هم با Group ID شناسایی می‌شوند. یعنی انسانها از نام گروه و لینوکس از GID هر گروه استفاده می‌کند. اگر هنگام ایجاد یک حساب کاربری جدید، هیچ گروهی به عنوان گروه پیش‌فرض آن حساب کاربری مشخص نشود، لینوکس یک گروه هم‌نام با یوزرنیم حساب کاربری ایجاد می‌کند و به آن یک GID اختصاص می‌دهد. در این بخش می‌خواهیم با ابزارهای مورد استفاده برای ایجاد و مدیریت گروه‌ها، بیشتر آشنا شویم.

پیدا کردن گروه پیش‌فرض یک حساب کاربری

ما می‌توانیم با استفاده از دستور getent (یا grep)، گروه پیش‌فرض یک حساب کاربری را پیدا کنیم. اگر به خاطر داشته باشید، چهارمین فیلد در فایل /etc/passwd، نشان‌دهنده‌ی GID گروه پیش‌فرض یک حساب کاربری بود. بیایید گروه پیش‌فرض حساب کاربری puppy را با هم مشاهده کنیم:

```
[root@localhost ~]# getent passwd puppy
puppy:x:1234:1234:Puppy Seeder:/home/puppy:/bin/zsh
```

همانطور که می‌بینید، گروه پیش‌فرض کاربر puppy، دارای GID برابر با ۱۲۳٤ می‌باشد.

برای این که بفهمیم نام گروهی که دارای GID برابر با ۱۲۳٤ می‌باشد چیست، کافی است به سراغ فایل /etc/group رفته و با استفاده از getent، به دنبال خطی که دارای عدد ۱۲۳٤ می‌باشد بگردیم. یعنی:

```
[root@localhost ~]# getent group 1234
puppy:x:1234:
```

همانطور که می‌بینید، با اجرای این دستور، خطی که در آن عدد ۱۲۳٤ وجود داشت به ما نشان داده شد. اگر به خاطر داشته باشید، فیلد اول این فایل نشان دهنده‌ی نام گروه بود، پس گروه پیش‌فرض کاربر puppy، برابر با puppy می‌باشد.

پیدا کردن گروه‌های یک کاربر با استفاده از *groups*

ما می‌توانیم با استفاده از دستور groups و ارائه‌ی نام حساب کاربری مورد نظر، کلیه‌ی گروه‌هایی که کاربر مشخص شده در آن وجود دارد را پیدا کنیم:

```
[root@localhost ~]# groups puppy
puppy : puppy
```

همانطور که می‌بینید، در خروجی این دستور، ابتدا نام حساب کاربری و پس از آن نام گروهی که کاربر درون آن عضو است نوشته می‌شود. در اینجا، می‌بینیم که کاربر puppy در یک گروه به نام گروه puppy عضو است.

اگر یک حساب کاربری در بیش از یک گروه عضو باشد، این گروه‌ها در یک خط و با یک فاصله‌ی خالی در کنار هم نمایش داده می‌شوند. برای مثال:

```
[root@localhost ~]# groups postfix
postfix : postfix mail
```

همانطور که می‌بینید، خروجی این دستور به ما می‌گوید که حساب کاربری postfix در دو گروه postfix و mail عضو می‌باشد.

ایجاد یک گروه جدید با *groupadd*

به طور کلی، برای این که یک کاربر را عضو یک گروه کنیم یا گروه پیش‌فرض یک حساب کاربری را تغییر دهیم، باید آن گروه از قبل وجود داشته باشد. برای ایجاد یک گروه جدید، از دستور groupadd استفاده می‌کنیم. فرض کنید میخواهیم یک گروه جدید به نام dandy درست کنیم. برای این کار:

```
[root@localhost ~]# groupadd dandy
```

همانطور که می‌بینید، استفاده از این دستور بسیار ساده می‌باشد. بیایید از ایجاد شدن این گروه اطمینان حاصل کنیم:

```
[root@localhost ~]# grep "dandy" /etc/group
dandy:x:1235:
```

اگربه خروجی این دستور نگاه کنید، میبینید که با این که با پسووردی روی گروه قرار ندادیم، اما در فیلد دوم خروجی، حرف x وجود دارد. برای این که مطمئن شویم که این گروه هیچ رمزی ندارد، باید به فایل /etc/gshadow مراجعه کنیم. ما با استفاده از دستور getent، در این فایل برای dandy جستجو می‌کنیم:

```
[root@localhost ~]# getent gshadow dandy
dandy:!::
```

همانطور که می‌بینید، در فیلد دوم خروجی، علامت ! وجود دارد. این یعنی روی این گروه، هیچ رمزی قرار نگرفته است.

نکته: به طور کلی، بهتر است که روی گروه‌ها پسووردی قرار ندهیم. قرار دادن پسوورد روی یک گروه باعث می‌شود که فقط کاربرانی که رمز آن گروه را دارند به آن گروه دسترسی داشته باشند. این یعنی کاربران باید رمز گروه را بین خود به اشتراک بگذارند که از نظر امنیتی، به اشتراک گذاشتن هر رمزی، بسیار اشتباه می‌باشد.

عضو کردن کاربران در یک گروه

پس از ایجاد یک گروه، می‌توانیم با استفاده از دستور usermod و آپشن‌های آن، کاربران را عضو آن گروه کنیم. برای مثال، بیایید کاربر puppy را عضو گروه dandy کنیم. برای این کار به صورت زیر عمل می‌کنیم:

```
[root@localhost ~]# usermod -aG dandy puppy
```

در این دستور، آپشن G-، کاربر puppy را عضو گروه dandy می‌کند و آپشن a-، باعث می‌شود که کاربر puppy، علاوه بر عضو شدن در گروه جدید، عضو گروه‌های قبلی خود نیز باقی بماند.

حال بیایید از عضویت کاربر در این گروه اطمینان حاصل کنیم:

```
[root@localhost ~]# groups puppy
puppy : puppy dandy
```

همانطور که می‌بینید، حال کاربر puppy عضو گروه dandy نیز شده است. لازم به ذکر است که اگر از آپشن a- دستور usermod استفاده نکرده بودیم، کاربر puppy از سایر گروه‌های خود خارج می‌شد و فقط عضو گروه جدید dandy می‌شد.

تغییر مشخصات یک گروه با *groupmod*

برای این که مشخصات یک گروه را تغییر دهیم، از دستور groupmod استفاده می‌کنیم. این دستور، آپشن‌های زیادی دارد که کاربردی‌ترین آنها، آپشن‌های g- و n- می‌باشند.

ما می‌توانیم با استفاده از آپشن g-، مقدار GID یک گروه را تغییر دهیم. بیایید GID گروه dandy را به مقدار ٤٣٢١ تغییر دهیم:

```
[root@localhost ~]# getent group dandy
dandy:x:1235:
[root@localhost ~]# groupmod -g 4321 dandy
[root@localhost ~]# getent group dandy
dandy:x:4321:puppy
```

همانطور که می‌بینید، با استفاده از آپشن g- دستور groupmod توانستیم GID گروه dandy را از ١٢٣٤ به ٤٣٢١ تغییر دهیم.

با استفاده از آپشن n- دستور groupmod، می‌توانیم نام یک گروه را تغییر دهیم. بیایید نام احمقانه‌ی گروه dandy را به group6 تغییر دهیم:

```
[root@localhost ~]# groupmod -n group6 dandy
[root@localhost ~]# getent group dandy
[root@localhost ~]# getent group group6
group6:x:4321:puppy
```

همانطور که می‌بینید، ما موفق شدیم با استفاده از آپشن n- دستور groupmod، نام گروه dandy را به group6 تغییر دهیم.

پاک کردن یک گروه با groupdel

برای پاک کردن یک گروه از دستور groupdel استفاده می‌کنیم. بیایید گروه group6 را پاک کنیم:

```
[root@localhost ~]# groupdel group6
```

همانطور که می‌بینید، پاک کردن یک گروه به همین سادگی می‌باشد. بیایید از صحت پاک شدن این گروه اطمینان حاصل کنیم:

```
[root@localhost ~]# getent group group6
```

همانطور که می‌بینید، دستور بالا چیزی در خروجی به ما نشان نمی‌دهد، این یعنی گروه group6 دیگر وجود ندارد.

نکته: پس از پاک کردن یک گروه، مهم است که در کل سیستم، به دنبال فایل‌ها و دایرکتوری‌هایی که متعلق به آن گروه بوده بگردیم و آنها را نیز پاک کنیم. ما می‌توانیم این کار را با دستور find انجام دهیم. برای مثال:

```
[root@localhost ~]# find / -gid 4321 2> /dev/null
```

دستور بالا، درکل سیستم به دنبال دایرکتوری و فایل‌هایی می‌گردد که مالکیت آنها در دست گروهی با GID برابر با ٤٣٢١ (همان گروه group6) باشد. 2> /dev/null بدین معنی است که کلیه‌ی پیغام‌های خطایی که در خروجی به ما نشان داده می‌شود، باید دور ریخته شوند. دلیل این امر، این است که هنگام اجرای دستور find، این دستور سعی به دسترسی به برخی از فایل‌های موجود در دایرکتوری /proc می‌کند. اگر صحبتهایمان در مورد FHS و وظیفه‌ی این دایرکتوری را بر عهده داشته باشید، میدانید که درون این دایرکتوری، اطلاعات پراسس‌ها ذخیره می‌شود. دلیل دریافت خطا در خروجی، این است که حین اقدام find به دسترسی به فایل‌های موجود در /proc، برخی از این پراسس‌ها terminate شده یا در حالت زامبی قرار می‌گیرند و در نتیجه، find به ما پیغام خطا می‌دهد. این امر برای ما مهم نیست و می‌توانیم پیغام‌های خطا را دور بریزیم. اگر به خاطر داشته

باشید، <2 عمل ری‌دایرکت کردن STDERR را برعهده داشت و /dev/null یک دیوایس فایل مخصوص بود که کلیه‌ی فایل‌هایی که به آن ارسال میشد را دور می‌ریخت.

Logها در لینوکس

در طول زمانی که سیستم‌عامل روشن و در حال کار می‌باشد، اتفاقات بسیار زیادی در حال رخ دادن هستند. برای این که بتوانیم یک سیستم را به درستی مدیریت کنیم، باید از کلیه‌ی اتفاقاتی که در پشت صحنه سیستم می‌افتد آگاه باشیم تا بتوانیم هنگام بروز یک مشکل یا خرابی، علت آن مشکل را پیدا کنیم. ابزار اصلی ما برای رسیدن به این هدف، استفاده از لاگ‌های سیستم می‌باشد.

کلیه‌ی توزیع‌های لینوکسی، از یک سرویس لاگ استفاده می‌کنند. لاگ‌ها پیام‌های کوتاهی هستند که اطلاعاتی نظیر آغاز یک رویداد و همچنین مکان و زمان آغاز آن رویداد را به ما می‌دهند. سرویس‌های لاگ، پیام‌های لاگ را به فایل‌ها یا حتی سرورهای جانبی ارسال می‌کنند. در صورت وقوع یک مشکل، ادمین سیستم می‌تواند با رجوع به این لاگ‌ها، مشکل را ریشه‌یابی کرده و سعی به رفع آن کند.

پروتکل Syslog

در اوایل، سیستم‌عامل یونیکس از روش‌های متفاوتی برای لاگ کردن رویدادهای سیستم استفاده می‌کرد. در واقع هر نرم‌افزار در این سیستم از یک روش متفاوت برای لاگ کردن استفاده می‌کرد و این امر کار را برای کاربران دشوار و عیب‌یابی را بسیار سخت می‌کرد. در اواسط دهه‌ی ۱۹۸۰، پروتکلی به نام syslog که برای نرم‌افزار sendmail ایجاد شده بود، تبدیل به یک استاندارد برای لاگ کردن سیستم و نرم‌افزارها در یونیکس شد. این امر، باعث شد که پروتکل syslog، در سیستم‌عامل لینوکس نیز خود را جا کند.

چیزی که پروتکل syslog را محبوب کرده است، استفاده از یک فرمت استاندارد برای هر پیام لاگ می‌باشد. این فرمت، زمان، نوع، شدت اهمیت و جزئیات یک رویداد را به صورت واضح، بیان می‌کند. علاوه بر این، هم سیستم‌عامل‌ها، هم نرم‌افزارها و هم سخت‌افزارها می‌توانند از این پروتکل استفاده کنند.

در لاگ‌های syslog، هر رویداد، یک نوع (type) دارد. نوع، مقداری است که موقعیت (facility) را مشخص می‌کند. موقعیت می‌گوید که چه چیزی (یکی از منابع سیستم، یک نرم‌افزار یا ...) پیام syslog را ایجاد کرده است. موقعیت می‌تواند مقادیر متفاوتی را داشته باشد. در جدول زیر، مقادیر متفاوت موقعیت (یا Facility Codeها) را مشاهده می‌کنیم:

جدول ۱۲- مقادیر موقعیت در syslog

کد	کلیدواژه	توصیف
0	kern	پیام توسط کرنل سیستم‌عامل ایجاد شده است.
1	user	پیام توسط رویدادی که کاربر شروع کرده ایجاد شده است.
2	mail	پیام توسط نرم‌افزار ایمیل ایجاد شده است.
3	daemon	پیام توسط سرویس‌های سیستمی که در پشت صحنه اجرا هستند ایجاد شده است.
4	auth	پیام‌های امنیتی و احراز هویت غیر سیستمی از این کد استفاده می‌کنند.
5	syslog	پیام توسط خود برنامه‌ی syslog ایجاد شده است.

6	lpr	پیام توسط پرینتر ایجاد شده است.
7	news	پیام‌ها توسط Network News Subsystem ایجاد شده است (سرورهایی که newsgroup را مدیریت می‌کنند).
8	uucp	پیام توسط برنامه‌ی UUCP (Unix-to-Unix Copy Program) ایجاد شده است.
9	cron	پیام توسط برنامه‌ی زمان‌بندی cron ایجاد شده است.
10	authpriv	پیام‌های امنیتی یا احراز هویت سیستمی از این کد استفاده می‌کنند. تفاوت اصلی آن با auth در این است که authpriv، پیام‌های احراز هویت را در یک فایل حفاظت شده ذخیره می‌کند.
11	ftp	پیام توسط سرویس ftp ایجاد شده است.
12	ntp	پیام توسط پروتکل ntp ایجاد شده است.
13	security	Audit Logها از این کد استفاده می‌کنند. Audit Logها لاگ‌های امنیتی می‌باشند که اتفاقات مهم را به ترتیب اتفاق آنها و به عنوان یک مدرک برای ارسال و دریافت آن پیام، ضبط می‌کنند.
14	console	پیام‌های هشدار از این کد استفاده می‌کنند.
15	solaris-cron	این کد، برای نوعی دیگر از برنامه‌ی زمان‌بندی استفاده می‌شود.
16-23	local0-local7	از این کد، برای پیام‌های محلی استفاده می‌شود.

همانطور که می‌بینید، syslog انواع متفاوتی از رویدادها را لاگ می‌کند. اما این پایان کار نیست؛ علاوه بر مشخص کردن نوع رویداد، پیام‌های syslog شدت هر پیام را نیز مشخص می‌کنند. شدت، به ما می‌گوید که هر پیام syslog، تا چه حد برای سلامت سیستم مهم است. در جدول زیر، مقادیر شدت موجود در syslog را مشاهده می‌کنیم:

جدول ۱۳- مقادیر شدت متفاوت در syslog

کد	کلیدواژه	توصیف
0	emerg	لاگی که این شدت را داشته باشد، نشان دهنده‌ی رویدادی است که باعث می‌شود سیستم غیرقابل استفاده شود.
1	alert	لاگی که این شدت را داشته باشد، نشان دهنده‌ی رویدادی است که نیاز به توجه فوری دارد.
2	crit	لاگی که این شدت را داشته باشد، نشان دهنده‌ی رویدادی است که مهم است اما نیازی به توجه و اقدام فوری ندارد.
3	err	لاگی که این شدت را داشته باشد، بیانگر به وجود آمدن یک خطا می‌باشد، اما خطایی که عملکرد سیستم یا نرم‌افزار را مختل نمی‌سازد و آنها می‌توانند به کار خود ادامه دهند.

4	**warning**	لاگی که این شدت را داشته باشد، بیانگر یک هشدار در مورد اتفاقی غیرطبیعی در سیستم یا یک نرم‌افزار می‌باشد.
5	**notice**	لاگی که این شدت را داشته باشد، بیانگر یک هشدار در مورد اتفاقی طبیعی اما مهم در سیستم یا نرم‌افزار می‌باشد.
6	**info**	لاگی که این شدت را داشته باشد، حاوی یک پیام دارای اطلاعاتی معمولی در مورد سیستم می‌باشد.
7	**debug**	لاگی که این شدت را داشته باشد، حاوی پیام‌های debug برای برنامه‌نویسان و توسعه دهندگان می‌باشد.

در پیامهای syslog، کد نوع، کد شدت و همچنین یک کامنت کوتاه در مورد وضعیت سیستم به ما نشان داده می‌شود. ما با استفاده از این اطلاعات، می‌توانیم اکثر مشکلات به وجود امده در لینوکس را حل کنیم.

پیدا کردن پیام‌های لاگ

به طور کلی، در اکثر سیستم‌های لینوکسی، فایل‌های لاگ در دایرکتوری `/var/log` قرار می‌گیرند. البته ممکن است برخی از فایل‌های لاگ توسط همه‌ی کاربران قابل مطالعه نباشند؛ که این امر به تمهیدات امنیتی موجود در هر سیستم بستگی دارد.

سرویس‌ها و برنامه‌های متفاوت لینوکسی، معمولا یک دایرکتوری مجزا برای خود درون دایرکتوری `/var/log` ایجاد می‌کنند و لاگ‌های مربوط به خود را درون آن قرار می‌دهند. برای مثال اگر روی سیستم سرویس `nginx` نصب شده باشد، لاگ‌های آن در `/var/log/nginx` قرار می‌گیرد.

از آنجایی که اکثر فایل‌های لاگ به صورت متنی می‌باشند، ما می‌توانیم با هر کدام از ابزارهای مشاهده متن، نظیر `cat`، `less`، `head`، `tail` و... برای مشاهده‌ی فایل‌ها استفاده کنیم و از ابزارهایی نظیر `grep`، برای فیلتر و جستجو برای عبارات مورد نظر، استفاده کنیم.

آشنایی با برنامه‌های لاگینگ در لینوکس

در طی سال‌ها، برنامه‌های بسیاری برای انجام لاگینگ در لینوکس ایجاد شده است. معروف‌ترین این برنامه‌ها به شرح زیر می‌باشند:

- **sysklogd**

اولین پروژه‌ی مربوط به پروتکل syslog در لینوکس می‌باشد. این برنامه از دو بخش تشکیل شده است: برنامه‌ی `syslogd` که سیستم را مانیتور می‌کند و برنامه‌ی `klogd` که فقط کرنل لینوکس را مانیتور می‌کند.

- **syslogd-ng**

این پروژه، ویژگی‌های پیشرفته‌تری نظیر ارسال پیام‌های syslog به یک سرور دیگر و... را دارد.

- **rsyslog:**

این پروژه، که حرف r موجود در ابتدای اسمش به معنای Rocket Fast (به سرعت یک موشک) می‌باشد، ادعا دارد که پرسرعت‌ترین برنامه‌ی لاگینگ می‌باشد. برنامه‌ی `rsyslogd` در طی سالیان

اخیر، از محبوبیت زیادی برخوردار شده و بسیاری از توزیع‌ها از این برنامه استفاده می‌کنند.

- **systemd-journald:**

این برنامه که بخشی از سیستم راه‌انداز systemd می‌باشد، امروزه توسط اکثر توزیع‌های لینوکسی به کار برده می‌شود. این برنامه از پروتکل syslog استفاده نمی‌کند و از روش کاملا متفاوتی برای لاگ‌کردن اتفاقات و رویدادهای متفاوت سیستم استفاده می‌کند.

ما در این بخش، به بررسی برنامه‌ی rsyslogd می‌پردازیم.

لاگینگ با استفاده از *rsyslogd*

همانطور که قبلا هم گفتیم، برنامه‌ی rsyslogd به صورت کامل از پروتکل syslog برای نوشتن لاگ‌ها استفاده می‌کند. اما ما چگونه می‌توانیم رفتار این برنامه در لاگینگ را بررسی کرده و لاگ‌های ایجاد شده توسط آن را مشاهده کنیم؟

تنظیمات اصلی برنامه‌ی rsyslogd، در فایل /etc/rsyslogd.conf قرار دارد. در برخی از توزیع‌ها، علاوه بر این فایل، فایل‌های موجود در دایرکتوری /etc/rsyslog.d که دارای پسوند .conf می‌باشند نیز عملکرد کلی این برنامه در مواجهه با لاگ‌ها را دیکته می‌کنند. این فایل‌ها، شامل قوانینی می‌باشد که به برنامه‌ی rsyslogd می‌گویند در برخورد با پیام‌های syslog که از سیستم، کرنل یا سایر برنامه‌ها دریافت می‌شوند، چه رفتاری از خود نشان دهد. فرمت کلی یک قانون در rsyslogd به صورت زیر می‌باشد:

```
facility.priority action
```

به طوری که:

- facility، نشان دهنده‌ی یکی از کلیدواژه‌های موقعیت (نوع) می‌باشد (کلیدواژه‌هایی که در جدول ۱۳ معرفی کردیم).
- priority، نشان دهنده‌ی یکی از کلیدواژه‌های شدت می‌باشد (کلیدواژه‌هایی که در جدول ۱٤ معرفی کردیم).
- action، مشخص می‌کند که rsyslogd هنگام برخورد با پیام syslog دریافتی، باید چه کاری انجام دهد.

هنگام استفاده از یک کلیدواژه‌ی شدت (یعنی بخش priority)، rsyslogd همه‌ی رویدادهایی که آن میزان شدت و همچنین همه‌ی رویدادهایی که میزان‌های شدت بالاتر از آن را (یعنی کد میزان شدت آنها از نظر عددی، کوچک‌تر) داشته باشند را لاگ می‌کند. مثلا، قانون:

```
kern.crit
```

کلیه‌ی رویدادهایی که در کرنل اتفاق می‌افتند را به شرط این که میزان شدت آن برابر با crit و همچنین alert یا emergency باشد لاگ می‌کند.

برای این که rsyslogd فقط و فقط رویدادهای دارای یک میزان شدت خاص را لاگ کند، باید قبل از کلیدواژه‌ی میزان شدت، از علامت = استفاده کنیم. یعنی:

```
kern.=crit
```

علاوه بر این، می‌توانیم از وایلدکاردها برای کلیدواژه‌ی موقعیت و همچنین شدت، استفاده کنیم. مثلا برای این که کلیه‌ی رویدادهایی که دارای میزان شدت emerg هستند را لاگ کنیم، می‌توانیم به صورت زیر عمل کنیم:

```
*.emerg
```

حال بیایید کمی بیشتر در مورد action صحبت کنیم. همانطور که گفتیم، action مشخص می‌کند که rsyslogd هنگام برخورد با یک پیام syslog، باید چه کاری انجام دهد. به طور کلی، شش action در rsyslogd وجود دارد:

- ارسال لاگ به یک فایل
- پایپ کردن لاگ درون یک برنامه
- نمایش پیام روی یک ترمینال یا کنسول
- ارسال پیام به یک سرور دیگر
- ارسال پیام به لیستی از کاربران
- ارسال پیام به کلیه‌ی کاربرانی که درون سیستم لاگین هستند.

بیایید به برخی از قوانین موجود در /etc/rsyslog.conf نگاهی بیاندازیم. در CentOS 7، فایل rsyslog.conf به چند بخش تقیسم شده است. ما به دنبال موارد نوشته در بخش #RULES هستیم:

```
[root@localhost ~]# cat /etc/rsyslog.conf
...
#### RULES ####
...
authpriv.*              /var/log/secure
...
mail.*                  -/var/log/maillog
...
cron.*                  /var/log/cron
...
*.emerg                 :omusrmsg:*
...
uucp,news.crit          /var/log/spooler
...
local7.*                /var/log/boot.log
```

قانون اول، کلیه‌ی پیام‌های دارای نوع یا کد موقعیت authpriv را، فارغ از میزان شدت آن، درون فایل /var/log/secure قرار می‌دهد. همانطور که می‌بینید، این قانون از وایلدکارد * برای مشخص کردن میزان شدت استفاده می‌کند. * به معنای کلیه‌ی میزان شدت‌ها می‌باشد. به طور کلی، این قانون کلیه پیام‌های امنیتی را درون فایل /var/log/secure قرار می‌دهد. از انواع اطلاعاتی که در این فایل ذخیره می‌شود، می‌توان به گزارش ورود یک کاربر به سیستم، خروج کاربر از سیستم، وارد کردن رمز اشتباه و... اشاره کرد.

قانون دوم، کلیه‌ی پیام‌های دارای کد موقعیت mail را فارغ از میزان شدت آن، درون فایل /var/log/maillog قرار می‌دهد. از انواع پیام‌هایی که درون این فایل قرار می‌گیرد می‌توان به پیام‌های مربوط به روشن شدن سرور ایمیل، خاموش شدن سرور ایمیل و... اشاره کرد. وجود علامت - در ابتدای /var/log/maillog، به این معنی می‌باشد که rsyslog فایل maillog را پس وارد کردن لاگ‌های

جدید درون آن، به صورت بلافاصله Sync نمی‌کند (روی هارددیسک ذخیره نمی‌کند). این امر سرعت rsyslog را بیشتر می‌کند، اما ریسک از بین رفتن لاگ‌های مهم در صورت خاموش شدن ناگهانی سیستم را نیز به همراه خود دارد.

قانون چهارم (*.emerg)، کلیه‌ی پیام‌هایی که دارای میزان شدت emerg باشند را، فارغ از نوع یا کد موقعیتشان، روی کنسول کلیه‌ی کاربرانی که در سیستم لاگین هستند، نشان می‌دهد. :omusrmsg: به معنای User Message Output Module می‌باشد و قرار گرفتن یک علامت * در کنار آن، به معنای نمایش پیام روی صفحه‌ی کنسول تک‌تک کاربران لاگین شده روی سیستم می‌باشد.

برای درک قانون آخر (local7.*)، باید مفهوم کد موقعیت local را به خاطر آوریم. ما گفتیم که از کدهای موقعیت local0 تا local7، برای پیام‌های محلی استفاده می‌شود. اگر یک برنامه‌نویس بخواهد برنامه‌ی rsyslogd، لاگ‌های ایجاد شده توسط برنامه‌اش را جمع‌آوری کند یا مثلا اگر بخواهیم لاگ‌های خاص یک برنامه را به سمت rsyslogd بفرستیم (مثلا لاگ‌های یک برنامه مثل squid و...)، می‌توانیم از این کدهای موقعیت استفاده کنیم. در اینجا، local7.*، کلیه‌ی اتفاقاتی که هنگام بوت سیستم می‌افتد را درون فایل /var/log/boot.log قرار می‌دهد.

ارسال لاگ‌ها به یک سرور جانبی با استفاده از *rsyslogd*

امروزه در بسیاری از دیتاسنترها، یک سرور لاگ مرکزی وجود دارد که لاگ‌های مربوط به کلیه‌ی دستگاه‌های موجود در شبکه را دریافت و آنها را برای کارهایی نظیر Monitoring ذخیره می‌کند. همانطور که قبلا هم گفتیم، ما می‌توانیم rsyslogd را طوری تنظیم کنیم که لاگ‌ها را به یک سرور دیگر ارسال کند. برای ارسال لاگ‌های سیستم به یک سرور، کافی است فایل rsyslog.conf را باز کرده، به انتهای فایل برویم و یک خط جدید را با محتوای زیر درون فایل قرار دهیم:

```
facility.priority       action
```

به طوری که facility، همچنان مشخص کننده‌ی کد موقعیت و priority همچنان مشخص کننده‌ی میزان شدت می‌باشد، اما action مشخص کننده‌ی مشخصات سروری است که می‌خواهیم لاگ‌ها به آن ارسال شوند. معمولا ما همه‌ی لاگ‌ها با هر کد موقعیت و هر میزان شدتی را به سرور لاگ ارسال می‌کنیم، پس معمولا از *.* برای مشخص کردن facility.priority استفاده می‌شود. اما مشخص کردن action برای ارسال لاگ‌ها به یک سرور جانبی، از فرمت زیر پیروی می‌کند:

```
TCP|UDP[(z#)]HOST:[PORT#]
```

همانطور که می‌بینید، مشخص کردن action به نظر گیج‌کننده می‌آید، پس بیایید آن را بیشتر توضیح دهیم:

- **TCP|UDP**: به معنای استفاده از پروتکل TCP یا UDP می‌باشد. ما می‌توانیم از پروتکل TCP یا UDP برای ارسال لاگ‌ها به سرور جانبی استفاده کنیم. ما در جلسه‌ی هشتم در مورد این دو پروتکل صحبت کردیم. به طور کلی، تفاوت بین این دو پروتکل این است که در صورت استفاده از UDP، احتمال از دست رفتن برخی از اطلاعات وجود دارد. پس اگر لاگ‌ها برایمان مهم هستند، باید از TCP استفاده کنیم. برای انتخاب UDP، از علامت @ و برای انتخاب TCP، از علامت @@ استفاده می‌کنیم.

- **[(z#)]**: وجود براکت‌ها ([]) به معنای اختیاری بودن استفاده از این آرگمان می‌باشد. z، به معنای استفاده از پروتکل zlib جهت فشرده‌سازی پیام قبل از ارسال آن به سمت سرور جانبی می‌باشد و #، مقداری عددی بین ۱ تا ۹ می‌باشد که میزان فشرده‌سازی را مشخص می‌کند. ۱ به معنای کمترین میزان فشرده‌سازی و ۹ به معنای بیشترین میزان فشرده‌سازی می‌باشد. توجه داشته باشید که برای فعال کردن فشرده‌سازی، z و همچنین میزان فشرده‌سازی باید حتما درون پرانتز قرار گیرند. برای مثال فشرده سازی با میزان ۵، باید به صورت (z5) نوشته شود.
- **HOST**: مشخص کننده‌ی نام دامنه (مثلا example.com) یا آدرس IP سرور جانبی می‌باشد.
- **[PORT#]**: وجود براکت‌ها ([]) به معنای اختیاری بودن استفاده از این آرگمان می‌باشد. این آرگمان، شماره‌ی پورتی که سرور جانبی روی آن برای پیام‌های لاگ گوش می‌کند را مشخص می‌کند.

بد نیست که با یک مثال، درک خود را از طریقه‌ی ارسال لاگ‌ها به یک سرور جانبی بهبود بخشیم. فرض کنید می‌خواهیم کلیه‌ی لاگ‌های سیستم خود را به یک سرور جانبی با آدرس آی‌پی ۱,۲,۳,٤ که روی پورت ۵۶۷۸ برای پیام‌های TCP گوش می‌کند، بفرستیم. برای این که در مصرف پهنای باند صرفه‌جویی کنیم، می‌خواهیم لاگ‌هایی که ارسال می‌کنیم در بالاترین میزان ممکن، فشرده‌سازی شوند. برای این کار، مقدار زیر را در انتهای فایل `rsyslog.conf` سیستم خود اضافه می‌کنیم:

```
*.*    @@(z9)1.2.3.4:56789
```

پس از اضافه کردن این خط، باید به سرویس rsyslog بگوییم که بار دیگر فایل تنظیمات خود را بخواند. ما این کار را با restart کردن این سرویس انجام می‌دهیم. در سیستم‌هایی که از سیستم راه‌انداز systemd استفاده می‌کنند، این کار را با دستور زیر انجام می‌دهیم:

```
[root@localhost ~]# systemctl restart rsyslog
```

ما در انتهای این جلسه، چگونگی ارسال لاگ‌های سیستم لینوکسی خود به یک سرور جانبی لاگ در ویندوز را در قالب یک تمرین، توضیح می‌دهیم.

ایجاد لاگ به صورت دستی با *logger*

برخی از اوقات، ممکن است بخواهیم خودمان یک لاگ را به صورت دستی ایجاد کنیم یا یک اسکریپت داشته باشیم که لازم داشته باشد وضعیت خود را به صورت لاگ روی سیستم ثبت کند. دستور logger، می‌تواند این کار را انجام دهد. برای استفاده از این دستور، کافی است logger را وارد کرده و سپس پیام مورد نظر خود را بنویسیم. مثلا:

```
[root@localhost ~]# logger Hahahaha I am laughing
```

برای مشاهده‌ی این پیام، کافی است به انتهای فایل /var/log/messages نگاه کنیم:

```
[root@localhost ~]# tail -1  /var/log/messages
Nov 18 13:04:27 localhost root: Hahahaha I am laughing
```

همانطور که می‌بینید، در این فایل، پیامی که با استفاده از logger ایجاد کردیم نوشته شده است.

اما این کل قابلیت‌های logger نیست. ما می‌توانیم با استفاده از آپشن p-، میزان شدت لاگ را مشخص کنیم. با استفاده از آپشن t- می‌توانیم لاگ ایجاد شده را با یک Tag خاص ایجاد کنیم تا پیدا کردن آنها ساده‌تر شود. برای مثال:

```
[root@localhost ~]# logger -p emerg -t mylog  Testing this!
[root@localhost ~]#
Broadcast message from systemd-journald@localhost.localdomain (Wed 2020-
11-18 13:16:15 +0330):

mylog[2694]: Testing this!


Message from syslogd@localhost at Nov 18 13:16:15 ...
 mylog:Testing this!
```

همانطور که می‌بینید، ما با استفاده از آپشن p- و سپس نوشتن کد emerg، به logger گفتیم که یک لاگ با میزان شدت emerg ایجاد کند. به دلیل این که لاگ ما میزان شدت emerg را داشت، پیام لاگ ما روی STDOUT نشان داده شد. ما با استفاده از آپشن t- و قرار دادن نام mytag، یک تگ به لاگ ایجاد شده‌ی خود دادیم و بدین صورت، پیدا کردن آن در فایل /var/log/messages را ساده‌تر کردیم:

```
[root@localhost ~]# tail -1 /var/log/messages
Nov 18 13:16:15 localhost mylog: Testing this!
```

همانطور که می‌بینید قبل از نشان دادن پیام نوشته شده توسط ما، تگ mytag قرار گرفته که ما را در پیدا کردن این لاگ، یاری می‌دهد.

Rotateکردن فایل‌های لاگ با *logrotate*

پرواضح است که روی سیستم‌های پرترافیک لینوکسی، حجم فایل‌های لاگ به سرعت بالا خواهد رفت. برای جلوگیری از این امر، بسیاری از سیستم‌های لینوکسی، از برنامه‌ی logrotate استفاده می‌کنند. این برنامه که روی اکثر سیستم‌های لینوکسی نصب می‌باشد، به صورت اتوماتیک فایل‌های لاگ ایجاد شده توسط rsyslog را با توجه به زمان ایجاد یا سایز آنها جدا کرده و آنها را آرشیو می‌کند. این فایل‌های لاگ آرشیو شده را می‌توان با استفاده از اعدادی که پس از نام آنها قرار می‌گیرد شناسایی کرد. برای مثال، بیایید به دایرکتوری /var/log رفته و فایل‌های لاگ مربوط به پروسه‌ی بوت را بررسی کنیم:

```
[root@localhost ~]# ls -l /var/log/boot.log*
-rw-------. 1 root root    0 Nov 16 10:34 /var/log/boot.log
-rw-------. 1 root root 8298 Nov  2 10:29 /var/log/boot.log-20201102
-rw-------. 1 root root 8866 Nov  3 12:26 /var/log/boot.log-20201103
-rw-------. 1 root root 8070 Nov  7 12:42 /var/log/boot.log-20201107
-rw-------. 1 root root 8850 Nov  8 10:26 /var/log/boot.log-20201108
-rw-------. 1 root root 8833 Nov 12 10:39 /var/log/boot.log-20201112
-rw-------. 1 root root 8056 Nov 15 12:42 /var/log/boot.log-20201115
-rw-------. 1 root root 8187 Nov 16 10:34 /var/log/boot.log-20201116
```

همانطور که می‌بینید، ما در اینجا یک فایل با نام boot.log داریم و سایر فایل‌ها، یک سری عدد که در واقع نشان دهنده‌ی تاریخ ایجاد هر فایل لاگ می‌باشند را در انتهای خود دارند. این فایل‌ها در واقع توسط logrotate، آرشیو یا rotate شده‌اند.

البته برنامه‌ی logrotate قابلیت‌هایی فراتر از آرشیو کردن فایل‌های لاگ دارد. این برنامه می‌تواند فایل‌های لاگ را فشرده‌سازی، پاک، یا در صورت نیاز، آن را به یک کاربر ایمیل کند. اما سوالی که پیش می‌آید این است که logrotate از کجا می‌داند که چگونه هر فایل لاگ را مدیریت کند؟

logrotate این کار را با نگاه کردن به فایل تنظیمات خود که در موقعیت /etc/logrotate.conf قرار دارد، انجام می‌دهد. بیایید نگاهی به محتویات این فایل بیاندازیم:

```
[root@localhost ~]# cat /etc/logrotate.conf
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# use date as a suffix of the rotated file
dateext

# uncomment this if you want your log files compressed
#compress

# RPM packages drop log rotation information into this directory
include /etc/logrotate.d

# no packages own wtmp and btmp -- we'll rotate them here
/var/log/wtmp {
    monthly
    create 0664 root utmp
      minsize 1M
    rotate 1
}

/var/log/btmp {
    missingok
    monthly
    create 0600 root utmp
    rotate 1
}
# system-specific logs may be also be configured here.
```

در چندین خط ابتدایی این فایل، تعدادی متغیر گلوبال وجود دارد. برای مثال، متغیر dateext را در نظر بگیرید. این متغیر به logrotate می‌گوید که باید تاریخ روز آرشیو فایلِ لاگ را به پایانِ اسمِ فایل‌هایِ لاگِ rotate شده اضافه کند (در مثال صفحه‌ی قبل، این موضوع را بررسی کردیم). اگر از متغیر dateext استفاده نکنیم، logrotate یک شماره به انتهای فایل‌های لاگ آرشیو شده اضافه می‌کند؛ مثلا:

boot.log.1
boot.log.2

پس از متغیرهای گلوبال، تنظیمات مخصوص آرشیو برای فایل /var/log/wtmp و سپس /var/log/btmp مشخص شده است. این تنظیمات، فقط روی این فایل‌ها اعمال می‌شوند و بر تنظیمات گلوبال، ارجعیت دارند.

اگر بار دیگر به محتویات logrotate.conf توجه کنید، می‌بینید که دقیقا پس از متغیرهای گلوبال، یک خط با عنوان زیر در آن وجود دارد:

```
include /etc/logrotate.d
```

این خط به logrotate می‌گوید که از فایل‌های تنظیمات موجود در دایرکتوری /etc/logrotate.d برای پیدا کردن دستورالعمل‌های اضافه در مورد چگونگی rotate کردن فایل‌ها، استفاده کند. فایل‌های تنظیمات موجود در logrotate.d، هم‌نام فایل‌های لاگی هستند که تنظیمات را برایشان می‌نویسیم. بیایید نگاهی به محتویات این دایرکتوری بیاندازیم تا این امر برایمان بهتر جا بیافتد:

```
[root@localhost ~]# ls /etc/logrotate.d/
bootlog  syslog  wpa_supplicant  yum
```

بیایید نگاهی به محتویات یکی از این فایل‌ها نیز بیاندازیم:

```
[root@localhost ~]# cat /etc/logrotate.d/bootlog
/var/log/boot.log
{
    missingok
    daily
    copytruncate
    rotate 7
    notifempty
}
```

همانطور که می‌بینید، در خط اول فایل‌های تنظیمات، موقعیت دقیق فایل لاگی که می‌خواهیم آن را rotate کنیم نوشته شده و پس از آن، متغیرهای مخصوص logrotate قرار گرفته‌اند. این متغیرها همان متغیرهایی که در فایل logrotate.conf دیدیم می‌باشند. پس بهتر است به توضیح جزئی این متغیرها بپردازیم. در جدول زیر، برخی از پرکاربردترین متغیرهای logrotate و مفهوم آنها را مشاهده می‌کنیم.

جدول ۱۴- پرکاربردترین متغیرهای logrotate

متغیر	توصیف
hourly	فایل لاگ را به صورت ساعتی rotate می‌کند.
daily	فایل لاگ را به صورت روزانه rotate می‌کند.
weekly *n*	فایل لاگ را به صورت هفتگی، در nاُمین روز هفته rotate می‌کند. مقدار ۰ برای n، به معنای روز یکشنبه، ۱ به معنای دوشنبه، ۲ به معنای سه شنبه و به همین ترتیب تا ۶، به معنای شنبه، می‌باشد. عدد ۷ برای n، یک مقدار ویژه است که به logrotate می‌گوید که فایل لاگ را هر هفت روز، rotate کند.
monthly	فایل لاگ را به صورت ماهانه rotate می‌کند.
size *n*	به جای rotate کردن بر حسب زمان، فایل‌های لاگ را با توجه به حجمشان rotate می‌کند، به طوری که n نشان دهنده‌ی حجمی است که logrotate را مجبور به rotate کردن فایل می‌کند. اگر پس از مقدار n هیچ واحدی قرار ندهیم (یا حرف K قرار دهیم)، logrotate مقدار n را با واحد کیلوبایت در نظر می‌گیرد. اگر حرف M یا G را پس از مقدار n قرار دهیم، logrotate عدد اختصاص یافته را به ترتیب مگابایت یا گیگابایت فرض می‌کند.

rotate *n*	فایل‌های لاگی که بیش از n بار rotate شده باشند را یا پاک می‌کند یا به یک کاربر دیگر ایمیل می‌کند (بستگی به سایر تنظیمات دارد). اگر n برابر با صفر باشد، فایل لاگ به جای rotate شدن، پاک می‌شود.
missingok	اگر فایل لاگ مشخص شده روی سیستم موجود نباشد، logrotate به ما پیغام خطایی نمی‌دهد و به کار خود ادامه می‌دهد.
notifempty	اگر یک فایل لاگ محتوایی نداشته باشد، logrotate آن فایل را rotate نمی‌کند.
compress	فایل‌های لاگ rotate شده را فشرده‌سازی می‌کند. این متغیر، به صورت پیش‌فرض از الگوریتم gzip برای فشرده‌سازی فایل‌ها استفاده می‌کند. برای این که الگوریتم فشرده‌سازی را تغییر دهیم، باید علاوه بر استفاده از متغیر compress، از متغیر compresscmd به علاوه‌ی نام الگوریتم فشرده‌سازی مورد نظر (xz، bzip2 و...) نیز استفاده کنیم. اگر بخواهیم برنامه‌ی فشرده‌سازی با یک آپشن خاصی اجرا شود، باید از متغیر compressoptions به علاوه‌ی متغیرهای قبلی، استفاده کنیم.

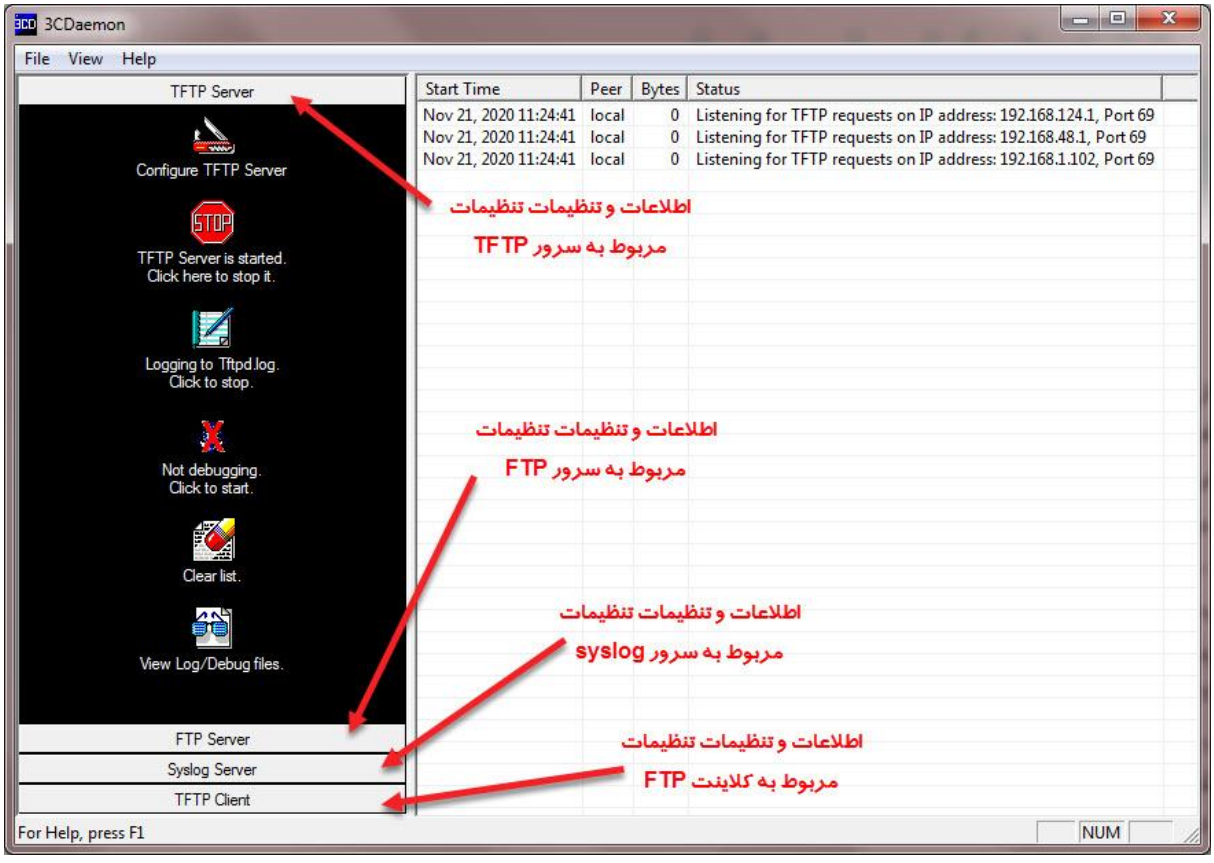
logrotate متغیرهای بیشتری نیز دارد که ما به توضیح آنها نمی‌پردازیم. در صورت تمایل، می‌توانید manpage مربوط به logrotate را مطالعه کنید.

تمرین: ارسال لاگ‌های سیستم به یک سرور لاگ جانبی

همانطور که گفتیم، در خیلی از مواقع، سرورها کلیه‌ی لاگ‌های خود را به یک سرور مجزا که فقط مخصوص جمع‌آوری لاگ‌ها می‌باشد ارسال می‌کنند. به طور کلی، برنامه‌های متفاوتی برای تبدیل یک سرور به یک سرور لاگ وجود دارد. برخی از این برنامه‌ها، برنامه‌های جامعی هستند که قابلیت‌هایی نظیر رسم نمودار، ارسال پیامک و ایمیل هنگام بروز مشکل و... را دارا می‌باشند و برخی دیگر، برنامه‌های بسیار کوچکی هستند که صرفا لاگ‌ها را از یک سیستم دریافت و آنها را در قالب یک سری فایل متنی، ذخیره می‌کنند.

در این بخش، ما با یکی از این برنامه‌های کوچک که صرفا لاگ‌ها را دریافت می‌کنند آشنا می‌شویم. فرض کنید می‌خواهیم کلیه‌ی لاگ‌های سرور لینوکسی خود را به یک سرور لاگ ویندوزی ارسال کنیم. برای این کار، ما نیاز به یک برنامه‌ی جمع‌آوری syslog برای ویندوز داریم. در اینجا، ما از برنامه‌ی 3CDaemon استفاده می‌کنیم که یک برنامه‌ی بسیار کوچک می‌باشد و فقط لاگ‌ها را دریافت و آنها را در یک فایل ذخیره می‌کند. این برنامه را می‌توانید از اینجا دانلود و نصب کنید. با توجه به این که نصب این برنامه نکته‌ی خاصی ندارد، به چگونگی نصب آن نمی‌پردازیم.

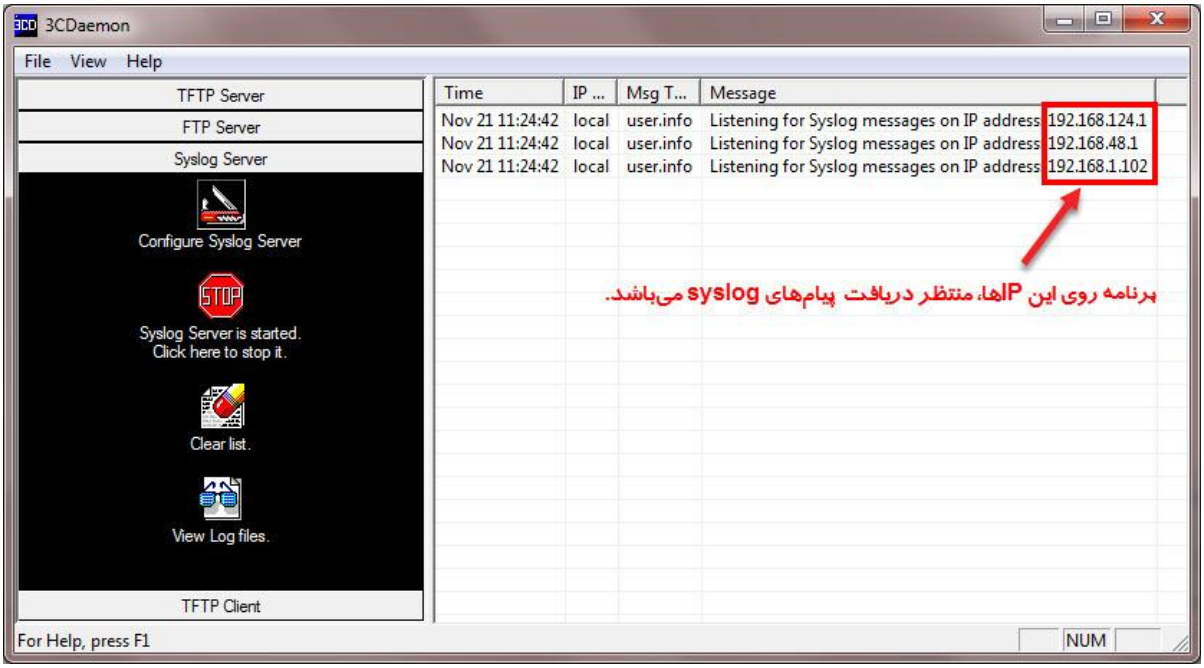
پس از نصب و اجرای برنامه، با نمایی نظیر تصویر زیر مواجه می‌شویم:



تصویر ٤- صفحه‌ی اصلی برنامه‌ی 3CDaemon

همانطور که می‌بینید، این برنامه علاوه بر عملکرد به عنوان یک سرور syslog، می‌تواند به عنوان یک سرور TFTP، FTP و همچنین به عنوان کلاینت FTP (یعنی اتصال به سایر سرورهای FTP و دریافت فایل از آنها) نیز عمل کند.

ما با بخش syslog server این برنامه کار داریم، پس روی آن بخش کلیک می‌کنیم:



تصویر ۵- بخش مربوط به سرور syslog برنامه‌ی 3CDaemon

همانطور که می‌بینید، اکنون این برنامه در حال عملکرد به عنوان یک سرور syslog می‌باشد و روی IPهایی که در تصویر مشخص شده، منتظر دریافت پیام‌های syslog می‌باشد. ما به تنظیمات 3CDaemon دست نمی‌زنیم و آنها را در حالت پیش‌فرض خود رها می‌کنیم.

همانطور که میدانید، برای ارسال لاگ‌های سیستم لینوکسی خود به یک سرور دیگر، باید تغییراتی در فایل تنظیمات rsyslog به وجود آوریم. در واقع، ما باید به rsyslog بگوییم که لاگ‌های دریافتی را به سمت یکی از آی‌پی‌هایی که 3CDaemon بر روی آن به دنبال پیام‌های syslog می‌باشد، ارسال کند. اما ما در تصویر ۳ آی‌پی می‌بینیم؛ پس rsyslog باید لاگ‌ها را به کدام آی‌پی ارسال کند؟

با توجه به این که سیستم لینوکسی و برنامه‌ی 3CDaemon در شبکه‌ی ۱۹۲.۱۶۸.۱.۰/۲۴ قرار دارند، ما باید فایل `/etc/rsyslog.conf` را ادیت کرده و به rsyslog بگوییم که کلیه‌ی پیام‌های دریافتی خود را به آی‌پی ۱۹۲.۱۶۸.۱.۱۰۲ ارسال کند (سایر آی‌پی‌های مشاهده شده در تصویر، بین سیستم لینوکسی و 3CDaemon مشترک نیستند و بدون روتینگ، به هم دید نخواهند داشت).

نکته: توجه کنید که این آی‌پی‌ها ممکن است در سیستم شما متفاوت باشند، پس ابتدا آی‌پی سیستم خود را چک کرده و بعد از آن به انجام تنظیمات بپردازید. دلیل این که به طریقه‌ی بررسی آی‌پی و شبکه‌ای که در آن هستیم نمی‌پردازیم، این است که در جزوه‌ی جلسه‌ی هشتم، این موارد را به طور کامل توضیح دادیم.

علاوه بر آدرس آی‌پی، بهتر است آدرس پورت را نیز مشخص کنیم. پورت مخصوص پروتکل syslog، ۵۱۴ می‌باشد. نکته‌ی قابلی توجه دیگر این است که 3CDaemon فقط پیام‌هایی که با پروتکل UDP ارسال شوند را دریافت می‌کند، پس باید به rsyslog بگوییم که لاگ‌ها را به پروتکل UDP به سمت 3CDaemon، ارسال کند.

خوب، با این اطلاعات، بیایید فایل `/etc/rsyslog.conf` را باز کرده، و به انتهای فایل، متن زیر را اضافه کنیم:

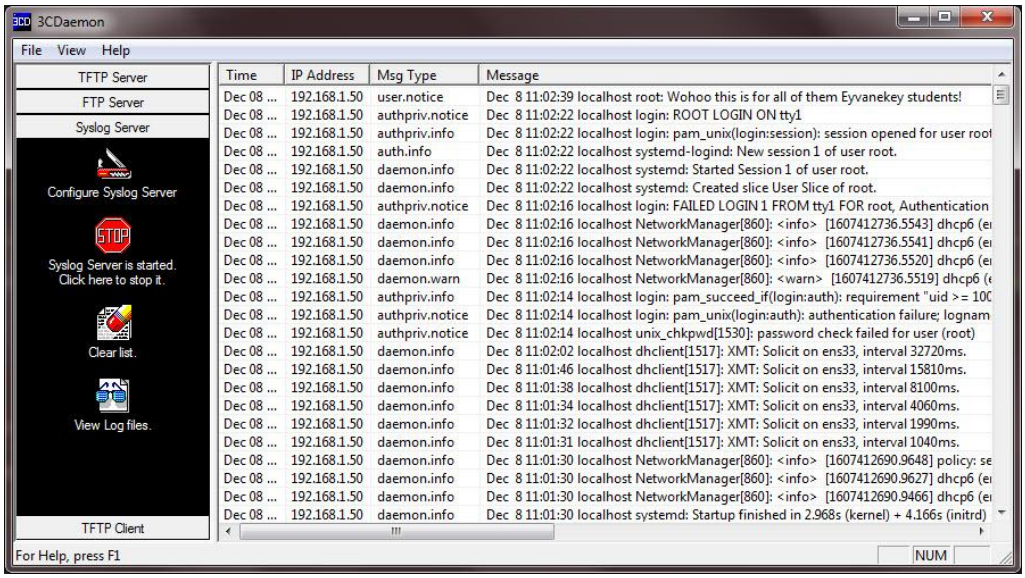
```
*.*                    @192.168.1.102:514
```

. به معنای ارسال همه‌ی لاگ‌ها، با هر کد موقعیت و هر میزان شدت می‌باشد. @ به معنای استفاده از پروتکل UDP جهت ارسال لاگ‌ها می‌باشد، 192.168.1.102 آدرس آی‌پی می‌باشد که سرور syslog روی آن قرار گرفته و 514 آدرس پورتی می‌باشد که سرور syslog در حال گوش کردن برای پیام‌های لاگ می‌باشد.

پس از وارد کردن این خط و ذخیره‌ی فایل، باید برنامه‌ی rsyslog را restart کنیم. پس:

```
[root@localhost ~]# systemctl restart rsyslog
```

به محض اجرای این دستور، صفحه‌ی 3CDaemon به سرعت توسط لاگ‌های سیستم پر می‌شود. حتی لاگ‌هایی که به صورت دستی با logger ایجاد می‌کنیم نیز در 3CDaemon قابل مشاهده می‌باشد:



تصویر ۶- صفحه‌ی 3CDaemon هنگام دریافت لاگ‌ها از سیستم لینوکسی