مفاهیم شبکه در سیستم عامل UNIX

گردآوری : وب سایت دانشجوی ایرانی

تاریخچه توسعه و تکامل یونیکس

تاریخچه طولانی یونیکس که می توان گفت بخشی از آن خوب و بخشی از آن بد می باشد , مطالعه ای است در پشتکار و استقامت, تغییرات ناگهانی در جهت توسعه نرم افزار, و جشن پیروزی تکنولوژی خوب بر روش برخورد هرج و مرج طلبانه ای که بعضی اوقات صنعت کامپیوتر با محصولات خوب پیش می گیرد . به دهه ۱۹۶۰ برمی گردیم , هنگامی که پیشرفت و توسعه کامپیوتر حوزه کاری تعداد اندکی شرکتهای پیشگام در صنعت و دانشگاههای پیشرو بود . در این دوران اغلب تحقیقات در آزمایشگاههای بل , ام آی تی , و جنرال الکتریک (تولیدکننده پیشروی کامپیوتر در آن زمان) صورت می گرفت . این سه کانون تحقیقاتی بر روی سیستم عاملی به نام MULTICS برای کامپیوتر بزرگ GE 645 تشریک مساعی کرده بودند. MULTICS پیروزی کاملی نبود, و بنابراین با شایستگی کنار گذاشته شد (به همین ترتیب تلاشهایی که بر روی کامپیوترهای GE صورت گرفت) . در ۱۹۶۹, کن تامپسون محققی در آزمایشگاههای بل و یکی از طراحان MULTICS , بازی Space Travel را برای MULTICS نوشته بود . در روزهای اشتراک زمانی, کاربران باید برای زمانی که بر روی کامپیوتر بزرگ صرف می کردند, هزینه می پرداختند . احتمالا Space Travel بازی خوبی بوده است, اما بر روی کامپیوتر بزرگ GE به خوبی اجرا نمی شد و برای بازی ۷۵ دلار هزینه در بر داشت . حتی برای یک آزمایشگاه تحقیقاتی نیز این مقدار اسراف وحشتناکی محسوب می شد, بنابراین کن تامپسون و همکارش دنیس ریچی

بازی را باز نویسی کردند تا بر روی کامپیوتر DEC PDP-7 که آن موقع در آزمایشگاههای بل بدون استفاده بود, اجرا شد . اما به منظور بردن Space Travel به DEC , تامپسون باید سیستم عامل جدیدی برای آن می نوشت و در همین نقطه بود که یونیکس امروزی ریشه گرفت .برایان کرنیگان دیگر محقق آزمایشگاههای بل به این سیستم عامل, لقب UNICS (Uuniplexed Information and computing System) را داد . سیستم عامل فوق در این راه به عنوان یونیکس شناخته شد و به کامپیوترهای قدرتمند تری منتقل گردید . در سال ۱۹۷۲, یونیکس به طرز دقیق بر روی ۱۰ کامپیوتر در حال اجرا بود, و در سال بعد تامپسون و ریچی, یونیکس را با زبان برنامه نویسی C بازنویسی کردند . زبان برنامه نویسی C قابل حمل تر بود و کمک کرد که یونیکس به سیستم عاملی تبدیل شود که می توانست بر روی انواع متفاوتی از کامپیوترها اجرا گردد .یونیکس به عنوان یک محصول توسط سرپرست حقوقیش AT&T دچار مشکل شد, چون دولت از معرفی محصولات کامپیوتری به صورت تجاری جلوگیری می کرد؛ این موضوع تا پیش از تجزیه AT&T سایر بازیگران را از صحنه بازار کامپیوتر خارج سازد. به دلیل تقاضا, AT&T اصلا یونیکس را به دانشگاهها, دولت و بعضی از شرکتها بخشید (با مطالبه قیمتی اسمی به منظور پرداختن هزینه مواد اولیه) . یونیکس در بین سالهای ۱۹۷۴ تا ۱۹۷۹ در واقع یک محصول تحقیقاتی بود که در دانشگاهها به منظور مقاصد آموزشی محبوبیت یافته بود . یونیکس ۲۹۷۴ شباهت چندانی به یونیکس ۱۹۷۹ نداشت؛ در طی این سالها برنامه های کمکی و ابزارهای بسیاری

به یونیکس افزوده شدند و در همان زمان نیز صنعت کامپیوتر به سرعت توسعه یافت و با این توسعه, بازار بالقوه یونیکس رشد پیدا می کرد . در این زمان توسعه یونیکس به آزمایشگاههای بل و AT&T محدود نمی شد . در سال ۱۹۷۴ دانشگاه برکلی کالیفرنیا کار خود را بر روی یونیکس آغاز کرد و Berkeley Software Distribution را تولید نمود . BSD شامل بسیاری از برنامه های کمکی متداول مانند ویرایشگر متن vi و پوسته C بود که می توانیم آنها را در یونیکس امروزی پیدا کنیم . تحقیق بر روی BSD تا امروز ادامه دارد . اما یکی از معماران BSD به نام بیل جوی, سان میکروسیستمز را پریشان و هیجان زده نمود و در آنجا کار تحقیقات و توسعه خود را بر روی یونیکس پیشرفت داد . امروزه سان یکی از رهبران بازار ایستگاه کاری یونیکس است (مجموعه SPARCstation این شرکت برای اغلب مردم مترادف ایستگاه کاری یونیکس می باشد), و SunOs تقریبا نگارش بسط یافته BSD است . چندین شرکت دیگر نیز از سان سرمشق گرفتند و نگارشهای مخصوص به خودشان از یونیکس را توسعه دادند . AIX از آی بی ام, Ultrix از DEC , و HP-UX از هیوات پاکارد, همگی مثالهایی از یونیکس نیمه اختصاصی می باشند . بعضی از نگارشها, مانند Coherent از Mark Williams Company , واقعا یونیکس نیستند, بلکه این نگارش فقط نرم افزاری است که مانند یونیکس واقعی به فرمانهای یونیکس پاسخ می دهد . در سال ۱۹۸۰ هنگامی که مایکروسافت نگارش کاهش یافته ای از یونیکس به نام زنیکس را منتشر ساخت, یونیکس در دسترس کاربران ریزکامپوترها نیز قرار گرفت . شاید مایکرسافت به

دلیل شکل دادن اتحادی با IBM مشهورتر باشد. این اتحاد (Disk Operating System) DOC مایکرسافت را به معروفترین سیستم عامل ریز کامپیوتر تبدیل ساخت. مسئوایت توسعه زنیکس به عهده شرکت سانتا کروز (شرکتی که بخشی از آن در مالکیت مایکروسافت است), گذاشته است و هم اکنون SCO UNIX محصول مهمی در دنیای یونیکس می باشد . پیش از سال ۱۹۸۳, یونیکس وضعیت مطلوبی نداشت زیرا به طور رسمی پشتیبانی نمی شد ؛ AT&T هیچ تضمینی درباره آینده یونیکس نمی داد . این وضع در سال ۱۹۸۳ تغییر کرد, AT&T در این سال نگارش Unix system V Release 1 را منتشر ساخت و قول داد نرم افزاری که برای این نگارش ایجاد می شود با تمام نگارششهای آینده یونیکس از AT&T سازگار خواهد بود . اما سایر نگارشهای یونیکس چه شدند ؟ در فرآیند توسعه یونیکس تعدادی ناسازگاری در نگارشهای گوناگون به وجود آمده بود که همین امر طراحی و توسعه یکنواخت نرم افزار را تا حدی مشکل می ساخت و می دانیم که هر چقدر هم سیستم عامل خوب باشد, اگر برنامه های کاربردی سودمندی وجود نداشته باشند, هیچ کس تصمیم به خریداری و استفاده از سیستم عامل را نخواهد گرفت . AT&T به همین دلیل, اتحاد بزرگ, یعنی UNIX System V Release 4 را معرفی نمود. این جدیدترین نگارش یونیکس, محبوب ترین و پراستفاده ترین فرمانهای SunOs, BSD, UNIX System V Release 3.2 و Xenix را ترکیب می کند . هم اکنون یونیکس در مالکیت ناول است که یونیکس را به عنوان بخشی از خرید UNIX System Labs به دست

آورد . از آنجایی که صنعت یونیکس به منظور تعیین استانداردها پیش می رود, یونیکس فراتر از کنترل یک شرکت است. حتی ناول نیز شعبه یونیکس را به گروه صنعتی X/Open واگذار نموده است . هنوز هم نگارشهای متفاوتی از یونیکس در سرتاسر بازار شناور هستند . سولاریس از سان را داریم که برای ایستگاههای کاری سان و کامپیوترهای شخصی مبتنی بر اینتل در دسترس است . Unixware از ناول را داریم ؛ و بسیاری از کاربران نیز در حال کشف قدرت یونیکس با لینوکس هستند, نگارشی با کاری مشابه یونیکس که به صورت مجانی در دسترس قرار دارد و برای کامپیوترهای شخصی نوشته شده است . به عنوان یک کاربر, مدیر سیستم یا طراح نرم افزار, نباید درباره این نگارشهای گوناگون و این که آیا مشکلاتی برای شما ایجاد خواهند کرد, نگران باشید . این نگارشهای مختلف در یک مورد اشتراک دارند و آن یک مورد پیروی از استانداردها می باشد . در واقع, امروزه نمی توانید سیستم عامل, محدودتری پیدا کنید که آینده اش در دست تعدادی خاص باشد . چرا استانداردها اهمیت دارند ؟ زیرا استانداردها تضمین می کنند برنامه ای کاربردی که برای بک نوع کامپیوتر طراحی می شود و از استانداردها پیروی می کند بر روی کامپیوتری از نوع دیگر نیز اجرا خواهد شد . گروههای مختلفی در شکل دهی استانداردهای یونیکس شرکت دارند : ANSI, UNIX System Labs (مؤسسه استانداردهای ملی آمریکایی که استانداردهای POSIX را توسط موسسه مهندسین برق و الکترونیک, IEEE , سرپرستی می کند) و X/Open .

چرا یونیکس ؟

یونیکس به دلایل گوناگونی به سیستم عامل بسیار محبوبی تبدیل شده است : یونیکس قابل حمل است . چون یونیکس به زبان برنامه نویسی C نوشته می شود و مقید به هیچ سخت افزار خاصی نمی باشد, تقریبا به هر نوع کامپیوتری برده شده است . کامپیوترهای شخصی, آمیگاها, مکینتاشها, ایستگاههای کاری, کامپیوترهای کوچک, کامپیوترهای بزرگ, و سوپر کامپیوتر ها از هر نوعی, سیستم عامل یونیکس را اجرا می کنند . این موضوع یعنی نرم افزاری که بر روی کامپیوتری نوشته می شود , به کامپیوتر دیگری قابل انتقال است . شرکتها دیگر به تولید کننده واحدی وابسته نمی باشند . آیا تا کنون عبارت فنی سیستمهای باز را شنیده اید ؟ از دیدگاههای مختلفی این عبارت مترادف یونیکس می باشد . در خرید سخت افزار و نرم افزاری که بایستی بتواند در طرح محاسباتی شما مجتمع گردد, آزاد هستید. بله, همیشه در هنگام آمیختن انواع و معماریهای سخت افزاری متفاوت مشکلاتی وجود خواهد داشت , اما حداقل لزومی ندارد که تمام تجهیزات خود را از تولیدکننده یا فروشنده واحدی خریداری کنید . چنین تولیدکننده ای می تواند با سخت افزار و نرم افزار اختصاصی, شما را در موقعیت سختی قرار دهد . نمی خرد . در عوض, سیستم عاملی را انتخاب می کنید که قادر به اجرای برنامه های کاربردی دلخواهتان باشد . یونیکس برنامه های کاربردی قدرتمندی را در تقریبا هر نوع نرم افزار از نشر الکترونیکی تا خوکارسازی اداری را دارا می باشد. یونیکس سیستم عاملی چندکاره می باشد . به طور همزمان بیش از

یک کاربر می توانند از سیستم یونیکس استفاده کنند . منابع سخت افزاری با ارزش مانند چاپگرها و سرویس دهنده های بزرگ توسط افراد بسیاری قابل استفاده می باشند. یونیکس سیستم عاملی چند وظیفه ای می باشد . می توانید دو وظیفه را به طور همزمان انجام دهید : به عنوان مثال می توانید در هنگامی که پست الکترونیکی خود را می خوانید, در پس زمینه پرونده ای متنی را قالب بندی کنید . در حین آنکه مفهوم چند وظیفه ای احتمالا آن قدری که برای نظریه دانان کامپیوتر اهمیت دارد, برای اغلب کاربران حائز اهمیت نیست, اما بعضی از اوقات عملی است که در اختیارداشتن آن مفید می باشد. شبکه سازی در یونیکس تعبیه شده است. امروزه یکی از بزرگترین چالشها, متصل ساختن انواع متفاوتی از کامپیوترها در سرتاسر نواحی کوچک و بزرگ می باشد . در یونیکس, شبکه سازی در برنامه ها و برنامه های کمکی گوناگونی تعبیه می شود .

سیستم عاملهای سازگار با یونیکس

برخی از سیستم های عامل سازگار با یونیکس در زیر معرفی شده اند :

Solaria محصول شرکت Sun Microsystems

IRIX محصول شرکت Silicon Graphics

AIX محصول شرکت IBM

HP-UX محصول شرکت Hewlen Packard

SCO-UNIX محصول شرکت Santa Cruz Operation.Inc

FreeBSD **نسخه رایگان یونیکس محصول دانشگاه برکلی آمریکا**

OpenBSD **محصول دانشگاه برکلی به همراه کدهای برنامه آن . این سیستم عامل با شعار « تلاش برای خلق امن ترین سیستم دنیا » ارائه شده است .**

Linux **سیستم عاملی است به همراه کدهای رایگان که ابتدا توسط یک دانشجوی فنلاندی به نام لینوس توروالدز نوشته شد و بعدا توسط شرکتهای مختلفی (با قیمت ناچیز) توسعه یافت . اسامی شرکتهایی که نسخه های لینوکس را تولید و عرضه کرده اند, عبارتند از :**

Readhat	Caldera
Corel	Debian
Slackware	Storm
SuSE	YellowDog
Turbo Linux	Mandarke

SunOS **نسخه قدیمی تر یونیکس که قبل از سولاریس توسط شرکت** Sun Microsystem **تهیه شده بود و هنوز در برخی از محیطها استفاده می شود. تمام این نسخه های متعدد, عظیم و جهانی از تفکر افرادی نشات گرفته اند که سی یال پیش در شرکت** AT&T **و در سکوت و بدون حمایتهای جهانی اقدام به پیاده سازی یونیکس کردند . بسیاری از آنها امروزه در دوران کهولت هستند و آن گذشته روشن را به دست فراموشی سپرده اند و برخی نیز در نقش پیش کسوتان و پدران معنوی هنوز در پالایش و رشد آن می کوشند . سیستم مدیریت فایل, فراخوانی روالهای سیستمی, فرامین و برخی از گزینه های پیکر بندی**

در نسخه های گوناگون یونیکس به روشهای متفاوتی پیاده سازی شده اند که این تفاوتها ناشی از نیاز تکنولوژیک روز بوده است ؛ ولی در مجموع خطوط اصلی این سیستم عامل توسط دو موسسه تعیین می شود :

گروه توسعه یونیکس در شرکت AT&T

گروه توسعه یونیکس BSD در دانشگاه برکلی

سولاریس, HP-UX از خطوط تعیین شده توسط AT&T تبعیت می کنند در حالی که FreeBSD , OpenBSD و لینوکس به گروه توسعه BSD گرایش دارند . سیستمهای عامل AIX و IRIX تلفیقی از نکات برجسته خط مشی هردو گرئه هستند لذا اندکی پیچیده تر به نظر می رسند.

معماری یونیکس

یونیکس مانند MS-DOS و OS/2 یک سیستم عامل است : نرم افزاری که کامپیوتر فیزیکی را کنترل و فرامین ما را تفسیر می کند .

سیستم عامل اعمال بسیاری انجام می دهد :

عملا برنامه را اجرا می کند . هنگامی که نام پرونده را در خط فرمان وارد می کنید, سیستم عامل با بارگذاری برنامه در حافظه کامپیوتر و اجرای آن کنترل را به دست می گیرد . تنظیمها در سیستم عامل می توانند در کارآیی واقعی برنامه ها اثر بگذارند. تمام ورودی و خروجی را در کامپیوتر کنترل می کند . هنگامی که پرئنده ای را حذف می کنید سیستم

عامل به کار خود ادامه می دهد و رکوردی از این پرونده را حذف می نماید . هنگامی که پرونده ای را ذخیره می کنید, سیستم عامل تضمین می کند که پرونده فوق بر روی پرونده موجودی نوشته نمی شود . سیستم عامل آنچه را که بر روی نمایشگر (یا نمایشگرها) نمایش می یابد, کنترل می کند و امکان می دهد که فرامین را توسط صفحه کلید یا ماوس وارد کنید . یونیکس این اعمال را از طریق سه بخش مجزا اما بسیار یکپارچه انجام می دهد : پوسته و هسته . هسته : مسئول تمام اعمال اولیه سیستم عامل می باشد . هسته در اولیه ترین سطح خود, وظیفه اداره حافظه کامپیوتر و چگونگی تخصیص دستورالعملهای نرم افزاری به این حافظه, اجرای تمام فرامین, سرپرستی سیستم پرونده, رسیدگی به خطاها و غیره را به عهده دارد . هنگامی که کامپیوتر روشن می شود, هسته شروع به کار می کند و بدون توجه به نرم افزار یا یا پوسته ای که اجرا می کنید, در حافظه باقی می ماند . به عنوان یک کاربر, نیازی نیست که درباره هسته فکر کنید ؛ تنها باید بدانید که هسته کار می کند . سیستم پرونده به پیگیری پرونده ها و محل قرارگیری آنها می پردازد . هر چیزی در یونیکس, خواه پرونده ای که در یک پردازشگر متن ایجاد می شود یا راه اندازی که برای فرستادن دستورالعملها به چاپگر استفاده می گردد, در پرونده ای شامل می شود . اگر زمینه داس یا مکینتاش را داشته باشید, متوجه شباهتهای زیادی بین زیرفهرستها ی داس و زیر فهرستهای یونیکس خواهید شد . پوسته یا مفسر خط فرمان, بخشی از یونیکس است که عملا در اغلب موارد در حال استفاده از آن خواهید بود . در اصل پوسته ها دستورالعملها را

می گیرند و آنها را به دستوراتی قابل فهم توسط هسته تبدیل می کنند . هنگامی که برنامه را اجرا می کنید, به پوسته می گویید که برنامه ای از هسته را اجرا نماید .

هسته سیستم عامل و پروسه ها

یونیکس از ساختار پیمانه ای (ماجولار) استفاده کرده است به گونه ای که در پائین ترین سطح یک هسته با بستر سخت افزار درگیر است و تمام برنامه ها به صورت هویتی مستقل حول هسته شکل می گیرند . در یونیکس هسته خودش یک برنامه اجرایی است که قبل از هر برنامه دیگر اجرا می شود و کنترل کل سخت افزار را به دست می گیرد و دسترسی پروسه ها به سخت افزار فقط از طریق هسته ممکن است . این هسته در سیستم عامل یونیکس Kernel نامیده شده است . هسته، قلب (و مغز) سیستم عامل محسوب می شود و هیچ برنامه ای بدون آن نخواهد توانست از مؤلفه های سخت افزاری سیستم استفاده نماید . برای شروع اجرای یک برنامه در محیط یونیکس، هسته یک پروسه ایجاد می نماید . پروسه برنامه ای در حال اجراست که دو قسمت زیر را در بر می گیرد :

کدهای اجرایی برنامه

حافظه اختصاص داده شده به آن برنامه به عنوان فضای کاری (Workspace) (تفاوت مفهوم برنامه – program – با پروسه – procces – از همین نکته ناشی می شود .)

برنامه – program – در حال اجرا مثل سرویس دهنده وب و پست الکترونیکی که فضا و منابع لازم را در اختیار دارند پروسه نامیده می شوند . پروسه ها را به صورت حبابهایی (یا

بادکنک) فرض کنید که بر روی هسته سیستم عامل تشکیل می شوند و تمام هستی و بضاعت خود را از هسته دارند و هر گونه اختلال در هسته آنها را نابود خواهد کرد . هسته با ایجاد پروسه ها این حبابها را خلق می کند و بر عملکرد آنها نظارت دارد : CPU را در اختیار آنها قرار می دهد به سادگی آنها را نابود می کند . محیط یونیکس محیطی چند وظیفه ای یا Multitask است و بالطبع در هر لحظه فقط یکی از پروسه ها، CPU را در اختیار دارد . پروسه های مختلف به صورت اشتراک زمانی (Time Sharing) از CPU استفاده می کنند و هسته به کمک سخت افزار، زمان در اختیار گرفتن CPU توسط یک پروسه را تنظیم و کنترل می نماید. هسته سیستم عامل همچنین فضای حافظه مورد نیاز را در اختیار پروسه ها قرار می دهد و باز هم به کمک سخت افزار جلوی تخطی پروسه ها از حریم فضای حافظه اختصاص یافته به آنها را در بر می گیرد . فضای حافظه اختصاص داده شده به هر پروسه، از دسترس پروسه های دیگر مراقبت خواهد شد و هر پروسه ای که سعی کند در خارج از فضای حافظه خود چیزی بنویسد یا بخواند توسط هسته سیستم عامل متوقف (و نابود) خواهد شد .

ساختار سیستم پرونده یونیکس

در سیستم عامل یونیکس هر چیزی به صورت یک پرونده تلقی و مدل می شود مثلا حتی دستگاههای جانبی مانند چاپگر یا پورتهای مخابره داده در قالب یک فایل مدل شده اد و برنامه نویس وقتی می خواهد اطلاعاتی را چاپ کند یا آنها را به منظور مخابره روی پورت

COM بفرستد، به سادگی فایل استاندارد آن را باز کرده ، درون آن می نویسد و سپس فایل را می بندد . این یکی از مفاهیم عالی یونیکس است که در سطح برنامه نویسی، برنامه نویس درگیر جزئیات سخت افزاری و نرم افزاری این ابزارها نخواهد شد بلکه این جزئیات در سطح سیستم عامل حل و فصل می شود . عناصر پروسه ها، صفها و بافرها نیز به صورت فایل مدل سازی شده اند . به بیان ساده، پرونده، ساختار کامپیوتر به منظور نگهداری اطلاعات می باشد . این اطلاعات در قالبی الکترونیکی قابل استفاده توسط کامپیوتر- بر حسب بیتها- ذخیره می گردند . بیت صفر یا یک است ؛ هنگامی که این بیتها در کنار یکدیگر قرار می گیرند، کاراکترهایی را شکل می دهند که من و شما قادر به تشخیص آنها می باشیم . در هر بایت هشت بیت، در هر کیلوبایت ۱۰۲۴ بایت، و در هر مگابایت ۱۰۴۸۵۷۶ بایت وجود دارد . البته نیازی به پیگیری تمام ۱۰۴۸۵۷۶ بایت در مستند یک مگا بایتی خود ندارید ؛ یونیکس نیز برای همین است . فقط باید با استفاده از فرمانی در یونیکس (مانند cat) یا برنامه ای کاربردی (واژه پرداز، صفحه گسترده، مدیر پایگاه داده ها، برنامه نشر رومیزی و غیره) پرونده را ایجاد و آن را نامگذاری کنید . هر پرونده ای دارای نام می باشد . نام پرونده حداکثر ۱۴ کاراکتر می باشد (در System V Release 4 طول اسامی پرونده می تواند نامحدود باشد، اما اگر ۱۴ کاراکتر اول در دو پرونده مجزا یکسان باشند، Release 4 اسامی این دو پرونده را یکسان در نظر می گیرد) . در سایر نگارشهای یونیکس، مانند BSD، می توان اسامی پرونده بسیار طولانی تر تا ۲۵۶ کاراکتر را داشت. در هنگام

نامگذاری پرونده ها در سیستم یونیکس رسومی وجود دارد . با وجود این که پسوندهای پرونده در یونیکس الزامی نمی باشند، اما این پسوندهای سه کاراکتری که توسط داس تحمیل می گردند، به منظور نشان دادن محتوی پرونده به استانداردی غیررسمی در یونیکس تبدیل شده اند . ساختار پرونده در یونیکس را می توانید به عنوان نوعی شجره نامه در نظر بگیرید . اگر بخواهیم فنی صحبت کنیم ، سیستم پرونده یونیکس، سیستم پرونده سلسله مراتبی نامیده می شود .

نکته :

از آنجایی که سیستم عامل یونیکس با زبان برنامه نویسی C نوشته شده است نسبت به بزرگی و کوچکی حروف حساس است .

در راس سیستم فایل یونیکس شاخه root **قرار دارد که به سادگی شاخه */* نامگذاری شده است و تمام شاخه ها و فایلها را در بر می گیرد. تمام زیر شاخه های دیگر (** Subdirectories **) درون این شاخه قرار می گیرند و کاربران فقط به برخی از آنان دسترسی خواهند داشت . عالیترین سطح دسترسی به سیستم فایل در اختیار کاربری است که ریشه (با تمام زیرشاخه های آن) تسلط دارد . در محیط یونیکس اگر چنین مجوزی داشته با شید با تغییر شاخه به شاخه / در راس سیستم فایل قرار خواهید گرفت که این کار با اجرای دستور زیر ممکن خواهد بود:**

**Ş cd **

هر فایل درون یکی از شاخه های این ساختار درختی ذخیره می شود و برای دسترسی به آن غیر از نام فایل باید سلسله مراتب شاخه های آن نیز مشخص باشد . مثلا اگر فایلی با نام hack.txt داشته باشید که در شاخه Usr ذخیره شده با آدرس usr/hack.tex قابل شناسایی و ذخیره و بازیابی است .

پوسته یونیکس

هنگامی که با سیستم یونیکس ارتباط برقرار می کنید، فورا به پوسته برقراری ارتباط خود می روید . اطلاعات مربوط به این پوسته، همچنین سایر اطلاعات برقراری ارتباط، و اطلاعات برقراری ارتباط تمام کاربران سیستم، در پرونده /etc/passwd قرار دارد (واضح است که این پرونده ابزاری برای مدیر سیستم می باشد، و برای کاربر مدام نمی باشد) . این پرونده بر حسب کاربر سازمان دهی می شود که در آن هر خط حاوی اطلاعات اولیه ای مانند، ID برقراری ارتباط و غیره درباره هر کاربر می باشد . آخرین فیلد در خط مربوط به شما، پوسته ای را که بعد از برقراری ارتباط قصد اجرای آن را دارید، فهرست می کند . این امر نقش بسیار مهمی در استفاده روزمره از یونیکس ندارد، اما نشان می دهد که هر جنبه ای از کاربرد یونیکس به نوعی توسط پوسته اداره می شود . پوسته مانند هر برنامه یونیکس یک برنامه می باشد . اما با مسئولیت بسیار مهم و منحصر به فردی تقویت می گردد : پوسته نقش میانجی بین را بین کاربر و سیستم یونیکس ایفا می کند . به کلامی دیگر, پوسته دستورالعملهای کاربر را به دستورالعملهای قابل رسیدگی برای سیستم یونیکس ترجمه می

کند . پوسته تمام جنبه های استفاده از یونیکس را آسان می سازد, به سیستم می گوید که پرونده های به خصوص را در کجا پیدا کند, فهرست شخصی شما در کجا قرار دارد و در کل چگونه با حضورتان به عنوان یک کاربر سروکار داشته باشید . این اطلاعات محیط یونیکس نامیده می شود, واژه ای که با شروع به تغییر دادن و پیشرفته ساختن استفاده خود از یونیکس مکررا با آن برخورد خواهید کرد . یونیکس, پوسته های بسیاری دارد . در اغلب موارد, تمایل بر این است که از پوشه انتخاب شده توسط مدیر سیستم استفاده کنید, زیرا این کار باعث به وجودآمدن سطحی از انطباق بین کاربران می شود . سایر کاربران در ایستگاههای کاری تک کاربره و ایستگاههای شخصی در انتخاب پوسته کمی از آزادی بیشتری برخوردار هستند . در اینجا برخی از پوسته های معروف توضیح داده می شوند :

Sh , پوسته Bourene . این پوسته ای است که در سال ۱۹۷۹ با نشر System V از یونیکس عرضه گردید, و از آن موقع تا کنون چندان تغییر نکرده است . این پوسته نام طراح خود, Stephen Bourne را یدک می کشد .

Ksh , یا پوسته Korn . این پوسته توسط David Korn در آزمایشگاههای بل طراحی گردید, پوسته Korn با افزودن خصوصیات مفیدی که ابتدا در پوسته C معرفی شدند (مانند تاریخچه کار و نامهای مستعار) بر مبنای عملکرد پوسته Bourne ساخته می شود . علاوه بر این دست نوشته ها و برنامه هایی که برای پوسته Bourne نوشته می شوند, بدون هیچ تغییری در پوسته Korn قابل استفاده هستند. Csh , پوسته C . این پوسته توسط بیل جوی

(بنیانگذار شرکت Sun Microsystems) در حالی که بر روی نگارش یونیکس برکلی (یا BSD) کار می کرد, طراحی گردید . پوسته C با معرفی تاریخچه کار, نامهای مستعار و سایر خصوصیات نسبت به پوسته Burn بهبود می یابد . این پوسته مانند زبان برنامه نویسی C , ساختیافته است . C و یونیکس تاریخچه مشترک مهمی دارند, خود یونیکس به زبان برنامه نویسی C طراحی شد و این عامل مهمی برای بردن آن به بسیاری از معماررییهای متفاوت سخت افزاری بود . اغلب طراحی برنامه یونیکس به زبان برنامه نویسی C صورت می گیرد .

Jsh , پوسته کار . گسترشی از پوسته Bourn می باشد . پوسته کار ابزارهایی تخصصی را برای رسیدگی به چندین کار شکل می دهد .

bash یا Bourn Again Shell . این پوسته توسط Free Software Foundation طراحی گردید . پوسته bash نیز با افزودن خصوصیات مفیدی که در پوسته های C و Korn یافت می شود, بر مبنای پوسته Bourn ساخته می گردد .

پروسه های کلیدی یونیکس

تمام پروسه هایی که به عنوان سرویس دهنده های کلیدی سیستم اجرا می شوند از عمومی ترین آنها یعنی سرویس دهنده وب تا کم کاربردترین آنها (مثل Character Generator) باید توسط پروسه های دیگر راه اندازی و فعال شوند . یعنی پروسه های سرویس دهنده به صورت مستقل و قائم به خود اجرا نخواهند شد بلکه پروسه های کلیدی دیگری آنها را راه

اندازی خواهند کرد . به عبارت دیگر هسته خود را درگیر اجرای پروسه های مختلف نمی کند بلکه چند پروسه مدیریتی و کلیدی را اجرا کرده و وظیفه فعال سازی و اجرای پروسه های سرویس دهنده را به این پروسه های کلیدی محول می نماید . این پروسه های کلیدی که در حین راه اندازی سیستم اجرا می شوند به شرح ذیل هستند :

پروسه init

این دایمون معمولا اولین پروسه ای است که پس از هسته, شروع به کار می کند و پدر تمام پروسه های کاربردی است که پس از راه اندازی سیستم, اجرا می شوند . این پروسه پس از پوت شدن سیستم, اسکریپتهای پیکر بندی پیکر بندی را اجرا می نماید تا راه اندازی سیستم با اجرا شدن پروسه های لازم تکمیل گردد . محل قرارگرفتن اسکریپتهای پیکربندی سیستم, به طور معمول /etc/init.d /etc/rc.d. **می باشد . بر اساس این فایلها پروسه های ثبت وقایع (**logger **), برنامه زمانبندی و پروسه های راه انداز واسطهای شبکه (**Network Interface **) پیکربندی و اجرا می شوند . همچنین** init **, متولی راه اندازی پروسه سرویس دهنده های مهمی است که پس از اجرا به صورت دایمون به یک شماره پورت خاص گوش می دهند, ترافیک بسته های** TPC **یا** UDP **را دریافت و پردازش می کنند و با کاربران در تمایل هستند . برخی از این دایمونها عبارتند از :**

Httpd **: سرویس دهنده وب که تقاضای** HTTP **و** HTTPS **را پردازش می کند و به طور معمول به پورت شماره ۸۰ گوش می دهد .**

Sendmail : سرویس دهنده پست الکترونیکی در محیط یونیکس که به پورت شماره ۲۵ گوش می کند .

NFS : پروسه سرویس دهنده ای که در محیط یونیکس فایلها را به اشتراک می گذارد . (Network File System) این سرویس دهنده در ابتدا توسط شرکت Sun Microsystems ابداع شد ولی بعدا تمام سیستمهای عامل سازگار با یونیکس از آن بهره گرفتند .

در ادامه این سه سرویس دهنده را اندکی بیشتر بررسی خواهیم کرد . این پروسه ها پس از اجرا, به شماره پورت مربوطه گوش داده و به حالت انتظار فرو می روند تا زمانی که تقاضای یک ارتباط دریافت شود یا ترافیکی به آن شماره پورت وارد گردد . دقت کنید که سه سرویس دهنده فوق دارای حجم ترافیک ورود بسیار بالایی هستند و باید به طور دائم در حافظه آماده پردازش این ترافیک باشند لذا احضار آنها روی سیستم فایل تاخیر زیادی به سیستم تحمیل خواهد کرد به همین دلیل این پروسه های سرویس دهنده, به طور مستقل توسط init در حین راه اندازی سیستم اجرا شده و به طور ثابت و دائم در حافظه باقی خواهند ماند . از طرف دیگر سرویسهای دیگری مثل TelNet یا FTP به طور مداوم مورد استفاده قرار نمی گیرند و ترافیک زیادی ندارند لذا به صورت دائم و مستقل به حافظه بار نخواهند شد . اگر تمام پروسه های سرویس دهنده ای که به یک شماره پورت خاص گوش می دهند به درون حافظه بار شوند و به طور مستقل و دائم به حالت انتظار بروند, کارآیی سیستم را کاهش داده و منابع آن با لاخص حافظه را تلف خواهند کرد . برای افزایش

کارآیی و صرفه جویی در منابع سیستم به غیر از پروسه های Httpd , NFS و Sendmail که مستقیما توسط init اجرا می شوند, هیچ پروسه ای نمی تواند به طور مستقل به یک شماره پورت گوش بدهد . در یونیکس پروسه ای به نام inetd (Internet Daemon) وجود دارد که به تمام شماره پورتهای TCP یا UDP گوش می دهد و در هنگام ورود ترافیک به هر یک از پورتها پروسه های متناظر با آن را فراخوانی کرده و اجرا می نماید .

پروسه inetd

دایمون inetd در مرحله راه اندازی سیستم توسط پروسه init به همراه سه پروسه NFS, Httpd و Sendmail اجرا خواهد شد . تنظیمات این پروسه حیاتی در فایلی با نام /etc/inctd.conf مشخص می شود . در این فایل شماره پورتهایی که inetd موظف است به آنها گوش بدهد, نوع پروسه متناظر با هر شماره پورت, نوع سرویس (UPD یا TCP) و نام هر سرویس در فایل /etc/services تعیین و پیکر بندی می شود. وقتی ترافیکی به ماشین وارد می شود, inetd ابتدا از طریق تنظیمات inted.conf سرویس مربوطه را مشخص کرده و سپس بر اساس فایل /etc/services پروسه متناظر با آن سرویس را راه اندازی و اجرا می نماید . پروسه سرویس دهنده پس از اجرا سرویس لازم را ارائه کرده و سپس خاتمه می یابد . در حقیقت به جای آن که پروسه های گوناگون به پورتها گوش بدهند, inetd به نیابت از همه آنها به تمام پورتهای تعیین شده گوش خواهد داد و در هنگام لزوم, آنها را راه

اندازی و اجرا خواهد کرد . به طور معمول inetd **سرویسهای زیر را راه اندازی و اجرا می نماید :**

Echo **: این سرویس دهنده هر رشته کاراکتری را که دریافت کند عینا به سوی مبدا آن باز می گرداند . این سرویس برای اشکال زدائی از شبکه مورد استفاده قرار می گیرد و لیکن ابزاری برای حملات** DoS **(به منظور اخلال در یک سرویس دهنده) خواهد شد . لذا موکدا توصیه شده که با تغییر در فایل پیکربندی** inetd **آن را غیر فعال نمائید .**

Chargen **: این سرویس دهنده یک فهرست از کاراکترها را تولید و برای مبدا ارتباط بر می گرداند . به طور معمول هدف این سرویس دهنده اندازه گیری کارآیی و سرعت یک سیستم در شبکه می باشد . حذف این سرویس دهنده چندان مهم نخواهد بود .**

FTPD **: این دایمون مهم سرویس دهنده** FTP **را راه اندازی می نماید تا کاربران راه دور از طریق آن به مبادله فایل بپردازند .**

TelNet **: این دایمون سرویس دهنده** TelNet **برای سرویس دهی از راه دور به کاربران است به گونه ای که بتوانند به سیستم وارد شده و فرامین مورد نظر خود را روی ماشین اجرا نمایند .(این سرویس دهنده نیز بسیار خطرناک است و اگر یک نفوذگر بتواند به نحوی در قالب یک کاربر مجاز به سیستم وارد شود, امنیت آن سیستم و حتی کل شبکه به خطر خواهد افتاد .**

Shell و login این دو پروسه نیز به کاربران اجازه می دهند که از راه دور به سیستم وارد شوند (با فرمان rlogin) یا یک نشست با سیستم ارتباط برقرار کرده و فرامین راه دور خود را از طریق برنامه rsh (remote shell) اجرا نمایند .

TFTP : این سرویس دهنده, پروسه TFTP را راه اندازی و اجرا می نماید . TFTP امکانات بسیار ساده و مختصری برای انتقال فایلها دارد .

برای آن که inetd را مجبور کنید تا یک پروسه سرویس دهنده را راه اندازی و اجرا نماید باید یک خط به فایل پیکر بندی inetd اضافه نمایید و در آن نام سرویس و پروسه اجرایی متناظر را تعیین کنید . به گونه ای که در این فایل مشاهده می شود برای غیر فعال کردن هر سرویس, کافی است درون فایل inetd.conf به دنبال خط مربوطه بگردید و با یک ویرایشگر ساده متن در ابتدای خط علامت # را قرار بدهید . inetd تمام خطوطی که با علامت # شروع شده باشند را خط توضیح تلقی کرده و نادیده می گیرد . در زیر مثالی از فایل پیکربندی inetd مشاهده می شود :

```
# These are standard services
#
ftp stream tcp nowait root / user/sbin/in.ftpd in.ftpd
telnet stream tcp nowait root / user/sbin/in.telnetd in.telnetd
#
shell stream tcp nowait root / user/sbin/in.rshd in.rshd
login stream tcp nowait root / user/sbin/in.rlogind in.rlogind
#exec stream tcp nowait root / user/sbin/in.rexecd in.rexed
```

هر کدام از فیلدهای مختلف هر سطر در فایل inetd.conf **مشخصه ویژه ای را تعیین و تنظیم می کنند که از چپ به راست آنها را معرفی می نمائیم :**

فیلد اول Services Name **: در این فیلد نام سرویسی که باید عرضه شود به صورت نمادین درج می شود . این نام کلیدی خواهد بود برای مراجعه به فایل پیکربندی** /etc/services **و استخراج تنظیمات سرویس دهنده و اجرای آن ؛ در حقیقت تنظیمات** /etc/services **شماره پورتهایی که باید** inetd **به آن گوش بدهد را مشخص می کند .**

فیلد دوم Socket Type **: این فیلد نوع ارتباط با سرویس دهنده را مشخص می کند . در این فیلد یکی از گزینه های زیر تنظیم می شود :**

stream **: سرویس مبتنی بر** TPC

Dgram **: سرویس مبتنی بر** UDP

Seqpacket **: سوکتهای مبتنی بر بسته های شماره گذاری شده**

Rdm **: سوکتهای مبتنی بر دریافت مطمئن پیام**

نکته ای مهم آن است که اگر ارتباط از نوع stream **تعیین شود دلیلی ندارد که لایه زیرین آن حتما** TCP **باشد و لیکن در شبکه اینترنت که مبتنی بر** TPC/IP **است سوکتهای نوع** stream **قطعا مبتنی بر** TCP **و سوکتهای** dgram **مبتنی بر** UDP **است .**

فیلد سوم Protocol Name **: در این فیلد نام پروتکل لایه انتقال درج می شود . در مورد سرویس دهنده های شبکه اینترنت, فقط گزینه های** tcp **و** udp **کاربرد دارند در حالیکه درج گزینه** `rpc/tcp` **(مخفف** remote procedure call **) و گزینه** rpc/udp **نیز ممکن خواهد بود .**

فیلد چهارم Wait Status **: در این فیلد دو گزینه حیاتی زیر می تواند درج شود :**

Wait **: این گزینه بدین معناست که یک پروسه سرویس دهنده قادر است فقط با یک بار اجرا چندین تقاضای همزمان را پردازش کند به عبارت دیگر تمام تقاضاها تحویل یک پروسه واحد و مستقل خواهد شد و لذا لازم نیست به ازای هر تقاضا یک پروسه مستقل و مشابه ایجاد شود .**

Nowait **: این گزینه مشخص می کند که** inetd **باید به ازای پذیرش هر تقاضا از یک شماره پورت, یک نسخه مستقل از یک پروسه سرویس دهنده را برای پاسخ به آن تقاضا اجرا کرده و پس از خاتمه سرویس آن را از بین ببر, لذا با ورود هر تقاضا پروسه از نو تولید و اجرا خواهد شد .**

فیلد پنجم User Name **: در این فیلد سطح دسترسی پروسه سرویس دهنده به منابع سیستمی تعیین می گردد, یک پروسه که در سطح** root **اجرا می شود بالاترین مجوز دسترسی به منابع سیستمی را خواهد داشت ولی هرگاه پروسه در سطح یک کاربر معمولی اجرا شود فقط منابعی را در اختیار خواهد داشت که برای آن کاربر تعیین شده است .**

نکته :

در سیستم عامل یونیکس, یک کاربر با نام root بالاترین سطح دسترسی را دارد و در حقیقت این کاربر کسی نیست مگر مسئول آن سیستم (یعنی Administrator)

فیلد ششم Server Program **: در این فیلد نام برنامه اجرایی که برای ارائه سرویس بدان شماره پورت باید اجرا شود, درج می شود .**

فیلد هفتم Server program Arguments **: وقتی برنامه مشخص شده در فیلد ششم اجرا شد لازم است یک سری از آرگومانهای ورودی جهت شروع به آن ارسال شود . این فیلد فهرست آرگومانهای خط فرمان را که باید به آن برنامه فرستاده شود, تعیین می کند .**

ارتباط بین پروسه های inetd,init **و پروسه های سرویس دهنده شبکه در شکل (۳-۳) نمایش داده شده است . به گونه ای که در این شکل دیده می شود پروسه** init **پروسه های دیگر را اجرا می نمایدکه مستقلا به شماره پورتهای مشخصی گوش می دهند . از بین این پروسه ها,** inetd **به مجموعه ای از شماره پورتها گوش داده و به ازای هر تقاضا, پروسه متناظر با آن را راه اندازی و اجرا می نماید . پروسه هایی مثل** httpd **یا** sendmail **به طور مستقل ترافیک خود را پردازش می نمایند در حالی که پروسه های دیگر همانند دایمون های** telnetd,ftpd **یا**tftpd **توسط** inetd **راه اندازی و اجرا خواهد شد .**

پروسه Cron

یکی دیگر از پروسه هایی که توسط هسته در حین راه اندازی سیستم فراخوانی و اجرا می شود, پروسه Cron نام دارد . این پروسه اجرای منظم و زمانبندی شده یک سری از فرامین (در زمانهای از قبل تعیین شده) را بر عهده دارد . (به عبارت دیگر این پروسه برای زمانبندی دقیق و منظم اجرای یک سری از پروسه های دیگر در خدمت مسئول سیستم خواهد بود) به عنوان مثال شاید, مسئول سیستم علاقه مند باشد که یک برنامه ویروس یاب در ساعت سه بامداد به جستجوی سیستم فایل مشغول شود یا نیمه شب هر شب از اطلاعات سیستم فایل نسخه پشتیبان تهیه گردد ؛ در این حالت او مجبور خواهد بود از پروسه Cron برای این کار بهره بگیرد . لذا به راحتی با تنظیم یک خط در فایل پیکربندی Cron که Crontab نام دارد هدف خود را دنبال می نماید . به طور معمول در سیستم عامل یونیکس این فایل پیکربندی در محلهای زیر ذخیره و بازیابی می شوند :

/ usr/lib/crontab یا etc/crontab/

هویت کاربر از نظر سیستم

برای ورود به یونیکس, هر کاربر باید یک حساب کاربری بر روی سیستم داشته باشد . هر پروسه نیز در یک سطح از دسترسی اجرا می شود و کاربر بر اساس مجوزهای آن سطح به منابع سیستمی دسترسی خواهد داشت . بدون داشتن یک حساب کاربری کسی قادر نخواهد بود به یونیکس وارد شده و سرویس بگیرد .

کاربری با نام root

قدرتمند ترین کاربر در سیستم عامل یونیکس با نام کاربری root تعریف شده است که بالاترین سطح دسترسی به منابع سیستم را خواهد داشت . کاربر root می تواند هر فایلی را بخواند, تغییر بدهد و حذف کند . بدین نحو او قادر خواهد بود پیکربندی کل سیستم را تنظیم کند, حسابهای کاربردی یا گروه ایجاد نماید . لذا در برخی از محاورات عمومی, کاربر root با عنوان Super user نامیده می شود . به طور معمول مشخصه کاربری (UID) چنین شخصی صفر است . مسئول سیستم زمانی که بخواهد تغییراتی در پیکره سیستم عامل ایجاد کند باید با این حساب کاربری به یونیکس وارد شود . (حساب root) هر گاه نفوذگر بتواند کلمه عبور کاربر root را به دست بیاورد یا به نحوی در سطح root به سیستم نفوذ کند هر کاری برای او ممکن خواهد بود .

حسابهای کاربری و گروهها (Account & group)

فایل etc/passwd

حسابهای کاربری در سیستم عامل یونیکس در فایل etc/passwd تعیین و نظیم می شود . در این فایل به ازای هر حساب کاربری یک سطر درج شده که مشخصات بسیار مهمی را از هر کاربر تعیین می نماید .

هر خط در این فایل شامل پارامترهایی است که توسط علامت : از هم تفکیک شده اند . این پارامترها از سمت چپ به راست به شرح زیر هستند :

Login Name **: در این فیلد نام حساب کاربری یا به تعبیری** UserID **درج خواهد شد . کاربر باید این نام را در هنگام** Login **وارد نماید . این نام محرمانه نیست و ممکن است همه آن را بدانند .**

Enerypted/Hashed Password **: در این فیلد کلمه عبور کاربر به صورت رمز نگاری شده درج می شود لذا حتی اگر این فایل در اختیار عموم قرار بگیرد, استخراج کلمه عبور به راحتی ممکن نخواهد بود . برای رمزنگاری کلمه عبور از روشهای متنوعی در یونیکس استفاده می شود (مثل :** Hash Algorithm **یا** encryption cipher **), وقتی کاربر کلمه عبور خود را وارد می کند . الگوریتم رمزگذاری بر روی آن اعمال شده و حالت رمزشده آن با این فیلد مقایسه می شود . اگر کلمه عبور پس از رمزگذاری با این فیلد یکسان بود به کاربر اجازه ورود به سیستم داده می شود . در غیر این صورت به او اجازه ورود داده نخواهد شد . اگر در فیلد کلمه عبور کاراکتر (** Σ **) درج شود بدین معناست که هیچ کس نمی تواند با** UserID **تعریف شده در فیلد قبلی به سیستم وارد شود .**

UID Number **: به هر حساب کاربری یک شماره شناسایی یکتا نسبت داده می شود . وقتی یک کاربر به یونیکس وارد می شود, این شماره منحصر به فرد ملاک شناسایی او خواهد بود و شطوح دسترسی به پروسه های اجرایی او بر اساس این شماره تعیین خواهد شد .**

Default GID number : در سیستم عامل یونیکس بعضی از کاربران در قالب یک گروه دسته بندی شده و برای همه آنها یک سطح دسترسی به فایلها و منابع سیستمی در نظر گرفته می شود . در این فیلد شماره گروهی که کاربر بدان متعلق است, درج می شود . (گروهها در فایل دیگر تعریف می شوند .)

GECOS Information : این فیلد با اطلاعات دلخواه کاربر پر می شود و به طور مستقیم مورد نیاز هیچ پروسه ای نیست . این اطلاعات شامل نام و نام خانوادگی یا شماره تلفن او می باشد.

Home Directory : در این فیلد نام شاخه ای درج می شود که کاربر پس از ورود به سیستم در آن شاخه قرار خواهد گرفت . به طور معمول این همان شاخه ای است که برای آن کاربر ایجاد شده و او فایلهای محلی خود را در آن ذخیره و بازیابی می نماید .

- Login Shell : در این فیلد نام برنامه ای درج می شود که به عنوان پوسته فرمان Command Shell- باید پس از ورود کاربر سیستم فرامین او را اجرا نماید .

فایل etc/passwd فایل متنی است ؛ هر کاربر با پروسه یا حتی یک نفوذگر قادر است فایل مربوطه را خوانده و کلمات عبور رمز شده را استخراج نماید . در این صورت نفوذگر بر اساس تکنیکهای شکستن کلمه عبور (Password Crack) سعی در کشف کلمات عبور کاربران می نماید لذا در برخی از گونه های یونیکس این فایل به صورت متنی ذخیره نمی شود بلکه کلمات عبور از درون فایل etc/passwd جدا شده و در یک فایل مجزا به نام

/etc/shadow یا /etc/secure ذخیره می شوند . لذا کاربران و پروسه ها به تمام فیلدهای فایل etc/passwd دسترسی دارند مگر به فیلد کلمه عبور ؛ هیچ کاربری مگر در سطح root قادر به دسترسی به فایل /etc/shadow نخواهد بود .

فایل /etc/group

از دیدگاه مدیریت سیستم تعیین مجوزهای دسترسی برای تک تک کاربران کار وقتگیری است. برای راحت تر کردن فرآیند تعرریف حسابهای کاربردی, تعریف شده و برای هر گروه مجوزهای دسترسی به دقت تعیین شود . سپس برای تعریف یک حساب کاربری می توان آن کاربر را عضوی از یک گروه تعریف کرد و بنابراین مجوزهای دسترسی او دقیقا مشابه تمام اعضای گروه خواهد بو.د . هر گروه در فایل /etc/group تعریف می شود . در این فایل به ازای هر گروه یک سطر وجود دارد که در زیر نمونه ای از محتویات آن دیده می شود :

Daemon:2:root,bin

Finance:x:25:alice,fred,susan

Hr:x:37:bob,mary

هر خط در این فایل شامل فیلدهایی است که به ترتیب از چپ به راست عبارتند از :

Group Name : در این فیلد نام یک گروه درج می شود .

Encrypted/Hashed Password : در این فیلد قاعدتا باید کلمه عبور گروه به صورت رمزنگاری شده درج شود ولی به طور معمول در آن کاراکتر `x` یا `*` قرار می گیرد و لذا هیچ کلمه عبوری برای گروه در نظر گرفته نخواهد شد .

GID Number : این فیلد توسط سیستم تنظیم می شود و شماره شناسایی یک گروه محسوب می شود .

Group Members : نام کاربردی تمام اعضای گروه در این فیلد درج می شود . هر نام توسط کاراکتر `,` از دیکری جدا می شود .

مثلا محتویات نشان داده شده فایل /etc/group تعیین می کند که : Fred,Alice و Susan در گروهی با نام finance تعریف شده اند و دارای GID شماره ۲۵ هستند .

مدیریت سیستم

اغلب وظایف اداره سیستم توسط کاربر ریشه یا کاربر برتر انجام می شوند . کاربران برتر امتیاز انجام هر کاری در سیستم را دارند : خواندن تمام پرونده ها, نوشتن در تمام فهرستها, و حذف همه چیز . این بخش آخری یعنی به عنوان کاربر ریشه می توانید به راحتی به خودتان صدمه وارد کنید . فرمانهای متداول کاربر برتر عبارتند از :

Date که تاریخ و ساعت سیستم را تنظیم می کند .

Init که وضعیت کاری سیستم را مثلا به وضعیت تک کاربره تغییر می دهد .

tar که به عنوان بایگان نوار عمل می کند .

Shutdown که کار سیستم را متوقف می سازد .

تمام کاربران می توانند Date و tar را اجرا کنند, اما فقط کاربر برتر می تواند از Date برای تغییر دادن ساعت سیستم استفاده کند . فرمانهای Init و Shutdown محدود به کاربر ریشه می باشند

کاربر ریشه مسئولیت افزودن کاربران جدید به سیستم و نصب سیستم عامل یونیکس را در بار اول به عهده دارد . متاسفانه هر نگارشی از یونیکس روش متفاوتی را برای نصب و افزودن کاربران جدید اتخاذ می کند و همین موضوع بسیاری از مدیران سیستم را به مشکل می اندازد.

این بخش فقط به آشکارترین اصول اولیه در مدیریت سیستم می پردازد .

برپاسازی سیستم جدید

در این فصل فرض می کنیم که محاسبات را بر روی سیستم یونیکس موجودی شروع کردید . اما مواردی وجود دارد که نیاز خواهید داشت سیستم خود را از آغاز برپا کنید . به عنوان مثال دوست دارید که ایستگاه کاری مستقل یونیکسی را برای اشفاده خودتان آماده کنید یا ممکن است که یک PC با نگارشی از یونیکس منتظر پیکربندی شخصی شما باشد . متاسفانه برای نصب نرم افزار یونیکس – خود سیستم عامل یا برنامه های کاربردی – یک راه معین وجود ندارد . بهترین سناریو, خریداری سیستمی با یونیکس از پیش نصب شده می باشد . اما در مورد نصب جدید, تمام آن چه واقعا می توانیم بگوییم این است که مسیرها

را دنبال کنید و خط تلفنی را برای پشتیبانی فنی باز نگهدارید . بعد از این که نگارش بخصوصی از یونیکس را نصب کردید, نوبت به جزئیات ساده مرتب نگه داشتن سیستم و محیط اجرای برنامه ها می رسد که دلیل وجود این فصل هم به همین خاطر است .معمولا بعد از نصب باید سیستم را خاموش و دوباره روشن کنید تا سیستم عامل امکان بارگذاری پیدا کند . هنگامی که سیستم عامل خود را در کامپیوتر بارگذاری می کند, مجموعه ای از پردازشها و بررسیهای سیستم را اجرا می نماید که پیغامهای رسیدگی که بر روی صفحه ظاهر می شوند, نشاندهنده این موضوع می باشند . اگر خطای مهیبی در طی نصب رخ ندهد, اعلان برقراری ارتباطی شبیه به این را خواهید دید :

Login: root

Password:

#

در طی نصب سیستم, احتمالا درخواست خواهد شد که کلمه عبوری را برای حساب کاربر ریشه ارائه کنید . اولین جلسه یونیکس خود را با وارد نمودن root **به عنوان نام برقراری ارتباط و فشردن کلید** Enter **بعد از کلمه شروع خواهید کرد . با این کار به عنوان کاربر ریشه با سیستم ارتباط برقرار می کنید (بسته به پیکربندی به خصوص سیستم ممکن است که تفاوت مشاهده شود .)**

چرا به عنوان کاربر ریشه وارد می کنیم ؟ در اینجا فرض می کنیم تا کنون هیچ نام برقراری ارتباطی در سیستم نصب نشده است . بنابراین نام برقراری ارتباط برای سیستم جدید کاملا

بی معنی می باشد .کاربر ریشه یا کاربر برتر به تمام بخشهای سیستم عامل یونیکس دسترسی دارد. کاربر توانایی خواندن تمام پرونده ها, دستیابی به تمام بخشهای سیستم پرونده و فرمانهای یونیکس را دارا می باشد . کاربر برتر تمام جنبه های استفاده از یونیکس و پیکربندی آن را کنترل می کند . به همین دلیل به عنوان کاربر ریشه وارد می شوید . به عنوان کاربر برتر اعلان ویژه ای به شما اعطا می گردد :

#

البته باید توجه داشته باشید که هیچ محدودیتی وجود ندارد, به عنوان کاربر ریشه می توانید به یونیکس خود صدمه جدی وارد کنید .

آماده سازی محیط

ماشین یونیکس شما در این نقطه مانند ذهنی است که هیچ باوری از طریق تجربه در آن حک نشده است : این به عهده شماست که با یاددادن نامش, ساعت و روز کنونی او را آموزش بدهید. ابتدا باید به دو روش سیستم را نامگذاری کنید : با نام سیستم و نام گره ارتباطاتی. در کل, نام سیستم برابر نگارش یونیکس مورد استفاده قرار داده می شود, در حالی که نام گروه ارتباطاتی نام منحصر به فردی است که اگر با سایر سیستمهای کامپیوتری در ارتباط باشید, برای شناسایی سیستم شما به کار می رود . با استفاده از فرمان setuname و گزینه های آن می توانید در یک زمان هر دو نام را تعیین کتید, در خط فرمان گزینه s- مشخص کننده نام سیستم و گزینه n- مشخص کننده نام گره می باشد :

 # setuname -n Attila -s SVR4

آنگاه باید زمان و تاریخ را تعیین کنید . این کار با استفاده از فرمان data **صورت می گیرد .**

با این فرمان باید دقیق باشید . شکل کلی آن شبیه به این می باشد :

 # data mmddttttyy

که در آن mm **به ماه ,** dd **به تاریخ ,** tttt **به زمان (البته بر طبق ساعت ۲۴ ساعتی نظامی) و** yy **به سال اشاره دارد . برای این که سیستم را با تاریخ ۱۲ آگوست ۱۹۹۴ در ساعت ۷ و ۲۶ دقیقه بعدازظهر تنظیم کنید, از خط فرمان زیر استفاده نما ئید :**

 # data 0812192694

wed aug 12 19:26:00 CDT 1994

در بعضی از کامپیوترهای شخصی ساعت به طور خودکار براساس اطلاعات پیکربندی PC **تنظیم می شود .**

وضعیتی که در آن قرار دارم

اغلب سیستمها در وضعیت چند کاربره راه اندازی می شوند که در این وضعیت سایر کاربران می توانند با سیستم ارتباط برقرار کنند و کل سیستم پرونده در دسترس می باشد (اصطلاح قابل دسترسی ساختن سیستم پرونده در قلمروی یونیکس سوارکردن سیستم پرونده می باشد) . اما این موضوع همیشه صحت ندارد (به عنوان مثال AU/X **اپل در وضعیت مدیریتی راه اندازی می شود و وانمود می کند که به سیستمی تک کاربره گرایش**

دارد) . در جدول زیر وضعیتهای اولیه سیستم را فهرست کرده ایم . اغلب مواقع تمایل خواهید داشت که در وضعیت چندکاربره کار کنید .

وضعیت	معنی
0	خاموش . سیستم خاموش است .
1	مدیریتی . کل سیستم پرونده در دسترس کاربر برتر می باشد, اما سایر کاربران نمی توانند با سیستم ارتباط برقرار کنند . وظایف پس زمینه امکان اجرا دارند .
2	چند کاربره . سایر کاربران می توانند با سیستم ارتباط برقرار کنند و به سیستم پرونده دستیابی داشته باشند .
3	RFS . اگر در حال سوار کردن سیستمهای پرونده از سایر ماشینها با شید, این وضعیت ماشینتان را به شبکه RFS مثل می کند و آن را برای تمام کاربران قابل دستیابی می سازد . این وضعیت, چند کاربره نیز می باشد .
S	تک کاربره . سیستم پرونده غیر قابل دسترسی است, سایر کاربران نمی توانند با سیستم ارتباط برقرار کنند, و تنها سیستم پرونده ریشه قابل دسترسی می باشد .

متوقف ساختن کار سیستم

قبلا گفتیم که 0 وضعیت خاموش می باشد . به کار بردن فرمان shutdown روش بهتی برای متوقف ساختن عمل سیستم است . اگر از shutdown به درستی استفاده شود, این فرمان سیستم را خاموش خواهد کرد و به سایر کاربران هشدار می دهد که سیستم در حال متوقف شدن می باشد:

 # shutdown

اگر به دلیلی پیغام خطایی را در هنگام اجرای این فرمانها دریافت می کنید, اسم کامل مسیر آنها را امتحان نمایید . در سیستمهای قدیمی تر (پیش تر از SVR4), خط فرمان به صورت زیر است:

 # /etc/ shutdown

در سیستمهای SVR4 , خط فرمان به شکل زیر است :

 # /usr/etc/ shutdown

بعد از به کاربردن فرمان shutdown , سیستم در مورد تصمیم به ارسال پیام برای کاربران پیش از متوقف ساختن کار سیستم و همچنین مدت زمان انتظار قبل از متوقف کردن کار سوال خواهد کرد . این فرمان در مورد متوقف کردن کار سیستم تایید نیز می گیرد . اگر به تنهایی بر روی ایستگاه کاری یونیکسی مشغول به کار می باشید, نیازی به فرستادن پیام نیست, و می توانید فورا کار سیستم را متوقف کنید . اگر در سیستمی چندکاربره کار می کنید و می خواهید هشداری به کاربران بدهید, می توانید پیامی را برای آنها بفرستید, و پیش از خاموش کردن سیستم ۶۰ ثانیه فرصت بدهید .

نکته :

فرمان **shutdown** مقداری از سیستمی به سیستم دیگر متفاوت است . اگر سیستم شما دقیقا بدین سبک عمل نمی کند, صفحات راهنمای حین کار یا مستندات را بررسی کنید .

در یونیکس system V می توانید فرمان shutdown را با استفاده از این پارامترهای خط فرمان سرعت بخشید :

 # shutdown -go -y

گزینه g- تاخیری را که پیش از متوقف کردن کار سیستماز روی لطف داده می شود, بر حسب ثانیه مشخص می کند . صفر (0) یعنی اصلا نمی خواهیم منتظر شویم . گزینه y– به سوال در مورد تصمیم واقعی در مورد خاموش کردن سیستم پاسخ بله می دهد . بنابراین برای عمل متوقف ساختن کار سیستم تاییدی گرفته نخواهد شد .

بعد از این که shutdown اجرا می شود چیزی شبیه به این نمایش می دهد :

Safe to power off

-or-

Press Any Key to Reboot

یا

Reboot the computer now .

بسیاری از سیستمهای جدیدتر یونیکس, مانند System V Release 4.2 که دارای ربط گرافیکی می باشند, با شمایلی روال shutdown را نیز ارائه خواهند کرد . در این موارد فرآیند متوقف کردن کار سیستم شامل بستن تمام پرونده های باز و آنگاه کلیک مضاعف بر روی شمایل متوقف کردن کار سیستم می باشد . در شکل ۱-۱۵ چنین شمایلی را می بینید . سوال خواهد شد که آیا قصد ذخیره کردن پیکر بندی فعلی سیستم را دارید (یعنی

آیا می خواهید تمام پنجره هایی که باز هستند در هنگام برقراری ارتباط مجدد نیز باز باشند) و آیا واقعا قصد متوقف کردن کار سیستم را دارید .

برقراری ارتباط

برقراری ارتباط روند ساده ای است که شما را به یونیکس معرفی می کند ؛ سیستم به منظور رسیدگی, با درخواست کلمه عبوری, پاسخ می دهد .

سیستم عبارت زیر را نمایش می دهد :

Login:

نام کاربری (که به عنوان نام برقراری ارتباط یا نام ورود نیز شناخته می شود) را وارد کنید و هنگامی که این عمل پایان رسید, کلید Enter را فشار دهید .

بعد از تایپ نام برقراری ارتباط, عبارت زیر نمایش خواهد یافت :

Password:

کلمه عبور خود را تایپ کنید . پایانه آن را تایپ می کنید, نمایش نمی دهد, و با این کار فرد دیگری نمی تواند با نگاه از پشت شانه های شما کلمه عبورتان را بدزدد .

اینها عناصر متداول در تقریبا هر سیستم مبتنی بر یونیکس می باشند . بسته به تولیدکننده یونیکس, اطلاعاتی را بر روی صفحه نمایش پایانه خواهید دید . اگر با نگارشی از یونیکس که توسطSanta Cruz Operation(SCO) توزیع شده, ارتباط برقرار کنید, توالی برقراری ارتباط به صورت زیر خواهد بود :

Welcome to SCO System V/386

Systemid!login: reichard

Password:

Welcome to SCO System V/386

From

The Santa Cruz Operation, Inc

پیغامهای دیگری را نیز ممکن است پیش از یک برقراری ارتباط موفق ببینید . اگر یونیکسی که استفاده می کتید در حال اجرای X Window System **باشد, صفحه ای با تعدادی پنجره و شمایل خواهید دید . اگر از نرم افزار پست الکترونیکی استفاده کنید, احتمالا پیغام زیر را خواهید دید :**

You have mail

علاوه بر این یونیکس برای تمام کاربران سیستم, news **را شکل می دهد . اگر برقراری ارتباط ناموفق باشد, پیغامی شبیه به این نمایش خواهد یافت :**

Login incorrect

Login:

اعلانهای سیستم

اگر برقراری ارتباط موفق باشد, علامتی شبیه به این به تنهایی یا به نوعی در داخل پنجره ای ظاهر می گردد :

$

به این علامت اعلان می گویند . همان طوری که انتظار دارید, اعلانهای $ مادگی برای دریافت فرمانها را نشان می دهند . اگر از دنیای DOS مده باشید, $ معادل اعلان C:> می باشد .

اگر کاربر ریشه بودید, اعلان شما بدین شکل می بود :

#

البته آن چیزی را که بلافاصله بعد از برقراری ارتباط می بینید, به سیستم و چگونگی پیکربندی ن بستگی خواهد داشت . یونیکس دو راه رابط گرافیکی و صفحه نمایش مبتنی بر متن را برای محاوره با کاربران در نظر می گیرد .

نکته مهمی وجود دارد که باید در مورد یونیکس و رابطها به خاطر بسپارید : در تمام مکانها قوانین یکسانی به کار می روند . برای اغلب متنها اهمیتی ندارد که از رابط متن یا گرافیکی استفاده می کنید . اگر بعضی از فرمانها هنگام اجرا در پنجره Xtream به تصویر کشیده شوند, خروجی این فرمانها هم یکسان خواهد بود و اهمیتی ندارد که ایا این فرمان در پنجره Xtream تحت X یا در یک پایانه مبتنی بر متن اجرا می گردند .

پوسته از طریق اعلان سیستم می گوید بسیار خوب, هر زمان که آماده هستید, می توانید فرمانی را وارد کنید . اعلان $ توسط پوسته های Korn و Bourn و اعلان ٪ توسط پوسته C استفاده می گردد . تا اینجا مهم است که بدانید پوسته مورد استفاده, نوع اعلان را تعیین می کند .

محیط متنی یونیکس

صفحه نمایش مبتنی بر متناز کاراکتر های الفبایی استاندارد و تعدادی خط برای محاوره با شما استفاده می کند . هیچ پنجره ای بر روی صفخه دیده نمی شود, و اصولا همیشه در هر زمان در حال انجام یک کار می باشید . سیستمهای قدیمی تر, صفحه نمایشهای مبتنی بر متن دارند . هنگامی که وظیفه ای به پایان رسید, صفحه به بالا می رود یا پاک می شودو اعلانی را نمایش می دهد . این اعلان به تنهایی در جایی از صفحه ظاهر خواهد شد (معمولا در نزدیک پایین متن موجود).

محیط گرافیکی یونیکس

رابط گرافیکی برای ارائه صورتی جذاب تر از گرافیکها, شمایلها و پنجره ها استفاده می کند برای اجرای برنامه, بر روی شمایلی کلیک می کنید یا می توانید فرمان استاندارد یونیکس را در پنجره ای وارد کنید . X Window System محبوب ترین رابط گرافیکی برای یونیکس می باشد . این رابط در ام آی تی طراحی گردید و هم به مجانی در دسترس است و فعلا توسط X Consortium, Inc سرپرستی می گردد . OSF/Motif , پیاده سازی ویژه ای از X می باشد که در Open Software Foundation طراحی گردیده است و به طور گسترده ای استفاده می شود . در این نمونه, اعلان در پنجره خودش, که معمولا بر چسب Xtream دارد, ظاهر می گردد.

مفهوم X Window System

X Window System – حتی مفهوم آن- اغلب کاربران جدید را سر در گم می سازد . آیا یک پوسته یونیکس می باشد ؟ خیر . آیا سیستمی گرافیکی هست که در آن همه چیز می رقصند و آواز می خوانند و تمام پریشانیهای محاسباتی شما را بهبود خواهند بخشید ؟ تا کنون خیر . آیا یک استاندارد است ؟ بله . آیا یک محیط کاری حرفه ای می باشد ؟ بله, اما به سختی .

X پیچیده و گیج کننده است, اما یک سیستم پنجره بندی گرافیکی نیز می باشد . X چندین پنجره (توسط چندین برنامه کاربردی اجرا می گردند)- ساده ترین و مهم ترین بلوکهای سازنده رابط گرافیکی کاربر (یا GUI که واژه ای معروف است)- را بر روی نمایشگری گرافیکی ارائه می کند .

X به عنوان رابط گرافیکی کاربر دو فایده آنی برای کاربر یونیکس دارد :

اول, X امکان می دهد که از صفحه نمایش خود بیشتر استفاده کنید, زیرا می توانید به جای ۲۴ خط در ۸۰ کاراکتر در حالت متن, به تمام نقاط موجود بر روی نمایشگر دسترسی داشته باشید . در طی سالیان اخیر, قیمت نمایشگرهای گرافیکی بسیار پایین آمده است, به طوری که اغلب کاربران می توانند از تواناییهای X بهره ببرند .

دوم, یک رابط گرافیکی کاربر حرکت انتقالی شما به یونیکس را ساده می کند و به کارهای روزانه تان سرعت می بخشد . مطالعات بسیاری نشان می دهند که یک رابط گرافیکی که به

شکل مناسبی طراحی می شود, مدت زمان آموزش را کاهش داده و بهره وری کارکنان را افزایش می دهد .

تقریبا هر بسته نرم افزاری مدرن یونیکس, از بازی xmahjongg تا واژه پرداز WYSIWIG (آن چه می بینید آن چیزی است که به دست می آورید) Asterix تحت رابط گرافیکی اجرا می شود . بر خلاف فرمانهای نقطه مرموز, FrameMarker,Island Write, و تقریبا تمام بسته های نرم افزاری مبتنی بر X, یک رابط گرافیکی منویی و دوستانه ارائه می کند . تمام این برنامه ها در بالای X Window System اجرا می گردند .

X سپس مبانی پنجره بندی گرافیکی را به روش قابل حمل به اکثر سیستمهای یونیکس فراهم می آورد . این رابط گرافیکی تقریبا بر روی تمام سیستمهای یونیکس, همچنین بر روی VMS, داس, ویندوز, مکینتاش اجرا می گردد و همین موضوعX را بسیار ویژه می سازد . Xتنها تلاش مدرن در جهت ارائه یک رابط گرافیکی کاربر نمی باشد, بلکه ویندوز مایکروسافت و سیستم عامل مکینتاش نیز دو رابط گرافیکی بسیار شناخته شده هستند .

چرا از X استفاده می کنیم ؟

انعطاف پذیری : X امکان که هر تعدادی رابط گرافیکی را بر روی سیستم پنجره ای زمینه لایه بندی کنید. می توانید برنامه های موتیف و Open Look, با برنامه هایی با سایر رابطهای مطلوب خود را اجرا نمایید . سیستمهای پنجره بندی اندکی این انعطاف پذیری را عرضه می دارند.

قابلیت حمل : برنامه های X بر روی گونه های گسترده ای از سیستمهای کامپیوتری اجرا می شوند . اگر ایستگاههای کاری یونیکس در شرکت یا دانشگاه شما متعلق به تولید کنندگان مختلفی باشد- اتفاقی بسیار متداول- آنگاه دانشی که از فراگیری X به دست می آورید, بر روی هر کدام از این سطوح کاری کمک بزرگی برایتان خواهد بود .

شفافیت شبکه ای : برنامه های X می توانند محاسبات را در سرتاسر شبکه انجام دهند . X Window System بر مبنای رابطه سرویس گیرنده/ سرویس دهنده محاسبات را به دو بخش عمده تقسیم می کند . این رابطه می تواند گیج کننده باشد, اما توانایی توزیع موثر برنامه های کاربردی از طریق شبکه را عرضه می دارد .

سرویس دهنده X

کلید درک X در فهمیدن مفهوم گیج کننده سرویس دهنده X می باشد .

در دنیاهای کامپیوتر های کوچک و ریزکامپیوترها, سرویس دهنده معمولا ابزاری سخت افزاری (VAX , AS/400 , یا سرویس دهنده پرونده ناول) می باشد که در مرکز شبکه ای کار می کند و وظیفه توزیع داده ها و قدرت پردازشی بین ایستگاههای کاری و پایانه های موجود در شبکه را به عهده دارد . چون سایر سیستمها در شبکه به صفحه نمایش شما دستیابی دارند, سرویس دهنده X را نمی توان مانند سرویس دهنده پرونده در شبکه ای محلی در نظر گرفت . در X , نقش سرویس دهنده (که بعضی اوقات به آن سرویس دهنده نمایش می گویند) معکوس می شود . سرویس دهنده برنامه ای است که بر روی ماشین

محلی اجرا می گردد و تمتم خروجیها را کنترل و آنها را بر روی نمایش ترسیم می کند . به ماشین محلی نمایش گفته می شود . این ماشین می تواند کامپیوتری شخصی در حال اجرای SCO Open Desktop یا اسپارک استیشن در حال اجرای Open Window باشد . سرویس دهنده, تصاویر را بر روی نمایشگر فیزیکی ترسیم, ورودی را از طریق صفحه کلید و ابزار اشاره گر (معمولا ماوس), دنبال, و پنجره ها را به شکلی مناسب به هنگام سازی می کند.

همچنین سرویس دهنده به عنوان پلیس ترافیک بین سرویس گیرنده های در حال اجرا بر روی سیستمهای محلی یا راه دور و سیستم محلی عمل می کند . سرویس گیرنده ها, برنامه هایی کاربردی هستند که وظایف خاصی را انجام می دهند (در X, اصطلاحهای سرویس گیرنده ها و برنامه های کاربردی به جای یکدیگر استفاده می شوند) . چون X محیطی شبکه ای می باشد, لزومی ندارد که سرویس دهنده و سرویس گیرنده لزوما بر روی یک ماشین محاسبه انجام دهند (اگر چه که در وضعیتهایی این کار را انجام می دهند) . X بدین طریق پردازش توزیع شده را مشخص می کند . به عنوان مثال, کامپیوتری شخصی که در حال اجرای SCO UNIX می باشد, می تواند قدرت پردازشی میزبان قدرتمندی در داخل شبکه را برای انجام کاری درخواست کند و نتایج محاسبات میزبان فوق بر روی نمایشگر PC نمایش یابند . در این مورد, سرویس گیرنده عملا در میزبان راه دور و نه در ماشین محلی اجرا می گردد و بنابراین پردازش در سرتاسر شبکه توزیع می شود . فکر نهفته در این کار ساده است ؛ محاسبات واقعی بایستی بر روی قوی ترین ماشین شبکه از لحاظ قدرت

محاسباتی صورت بگیرد و لزومی به انجام آن بر روی کامپیوتر مورد استفاده کاربر نیست. اغلب کاربران X به عنوان کمکی برای کنترل نمایش خود مدیر پنجره ای را اجرا می کنند . مدیر پنجره برنامه ای است که چگونگی ظاهر شدن و عمل کردن رابط (ظاهر واقعی برنامه) بر روی صفحه تصویر را بیان می کند . X ظاهر و حسی (ترتیب خاص عناصر – نوارهای پیمایش, نوارهای عنوان و غیره – بر روی صفحه تصویر) را برای این پنجره ها ارائه نمی کند . همان طوری که قبلا تذکر داده ایم, X بلوکهای سازنده رابطی گرافیکی را فراهم می آورد و کاربر در لایه بندی هر ظاهر و حسی در بالای X آزاد است. از آن جایی که X Window System رابط کاربری را در اختیار ندارد, در لایه بندی ظاهر و حس بخصوصی بر روی آن آزاد می باشید . و در واقع اغلب کاربران به طور مستقیم با X سروکار ندارند, بلکه بیشتر با راه حلهای ارائه شده توسط تولید کننده یا فروشنده سر و کار دارند و در اینجاست که موتیف و Open Look وارد می گردند .

راه اندازی X

پیش از این که بتوانید هر برنامه کاربردی X را اجرا کنید, باید سرویس دهنده X را راه اندازی کنید . سرویس دهنده X مسئولست کنترل نمایش – صفحه کلید, ابزاری اشاره گر (معمولا ماوس), و حداقل یک نمایشگر تصویری یا در بعضی از سیستمهای چند سره بیشتر از یک نمایشگر تصویری را به عهده می گیرد .

سرویس دهنده X به تنهایی ارزش چندانی ندارد ؛ تمام آن چیزی که به دست می آورید یک الگو و مکان نمای X است . در هنگام راه اندازی سرویس دهنده X تمایل خواهید داشت که تعدادی از برنامه های کاربردیX , از جمله مدیر پنجره را نیز راه بیاندازید . برنامه های کاربردی اهمیت دارند؛ سرویس دهنده X فقط زیر ساختار را فراهم می آورد . متاسفانه برای راه اندازی سرویس دهنده X راههای بی شماری وجود دارند . بسیاری از تولید کنندگان یا فروشندگان ایستگاههای کاری, در تلاشی بیهوده به منظور افزودن ارزش و گیج کردن شما, روش راه اندازی X را اختصاصی نموده اند.در حالی که اگر ایستگاههای کاری را فقط از یک فروشنده یا تولید کننده خریداری کنید, ممکن است این خصوصیات ارزش افزوده به نظر آیند, اما وقتی سعی کنید که فرمانهای مرموز و اختصاصی هر سیستم را فرا بگیرید, این خصوصیات به ارزش کاهش یافته تبدیل می گردند . نکته اصلی در مورد یونیکس این است که سیستم عامل فوق بر روی تمام سطوح کاری یا سیستمها به طور نسبی عمل می کند . این نکته در مورد X نیز صحت دارد . به هر حال اگر سستم شما از قبل برنامه ای برای راه اندازی سیستم X دارد, اجازه استفاده از آن را دارید . در غیر این صورت می توانید سرویس دهنده X را با فرمانهای xinit یا startx راه اندازی کنید .

نکته :

اگر قبلا **X** را اجرا کرده اید, یعنی صفحه نمایشی گرافیکی در جلوی خود دارید, نیازی به اجرای **xinit** نخواهید داشت زیرا این کار (یا هم ارز آن بر روی سیستم مورد استفاده) قبلا برایتان انجام شده است .

معمولا xinit **می تواند بدون هیچ گونه آرگومانی اجرا شود :** $ xinit

xinit **سرویس دهنده** X **را راه اندازی می کند . سرویس دهنده** X **برنامه ای به نام مناسب** X **می باشد که معمولا در** /usr/bin/X11/ **ذخیره می گردد .** xinit **بعد از راه اندازی سرویس دهنده** X **, برنامه های فهرست شده در پرونده** xinitrc **. را اجرا می کند . این پرونده بسیار شبیه به پرونده های** profile. **یا** login. **کد مورد استفاده** sh **و** csh **می باشند, در فهرست شخصی کاربر قرار دارند . پرونده** xinitrc **. برنامه هایی را که می خواهید بعد ازشروع به کار** X **راه اندازی شوند, فهرست می کند . این برنامه ها معمولا شامل ساعت کار گرافیکی, حداقل یک** xterm**, و مدیر پنجره ای, مانند** olwn,twm **یا** mwm **می باشند .**

اگر xinit **نتواند پرونده** xinitrc **را پیدا کند, فقط سرویس دهنده** X **و یک پنجره ۲۵ خطی** xtrrm **را راه اندازی می نماید .**

استفاده از startx

راز بزرگی در فرمان startx **وجود ندارد . این فرمان فقط فرمان** xinit **را تحت پوشش فراخوانی می کند . اغلب پیاده سازیهای** X Window **در** PC **مبتنی بر یونیکس, مانند**

Xfree86 , به منظور شروع سرویس دهنده X از startx استفاده می کنند اما در سایر موارد, فرآیند پیکر بندیxinit (مانند پیکر بندی پرونده xinitrc) مانند فرآیند مورد استفاده startx می باشد .

استفاده از ماوس

X Window System احتیاج به ماوس دارد ؛ با وجود ابن که می توان در داخل X بعضی از کارها را از صفحه کلید انجام داد, اما در کل برای انجام کارهای اساسی نیاز به ماوس دارید . ماوس امکان می دهد که به طور مستقیم اشاره گری را بر روی صفحه تصویر کنترل کنید . این اشاره گر هنگامی که بین پنجره ها, شمایل ها, جعبه های محاوره, منوهای گزینشی و سایر عناصر صفحه تصویر حرکت می کنید به عنوان مامور شما عمل می کند . X فرض می کند که ماوس مورد استفاده سه دکمه ای است, در حالی که ماوسهای تا پنج دکمه را نیز پشتیبانی می کند (بله ماوسهایی تا پنج دکمه و بیشتر نیز وجود دارند, اما کاربرد آنها بیشتر به حوزه CAD/CAM محدود می گردد) . X به طور داخلی به دکمه های ماوس سه دکمه, با عنوان Button2, Button 1 و Button3 رجوع می کند .

استفاده از ماوس برای حرکت در محدوده صفحه تصویر کار بزرگی نیست, به خصوص اگر که قبلا با ویندوز مایکروسافت, os/2, یا سیستم عامل مکینتاش کار کرده باشید . اما اگر به کار در محیط گرافیکی با ماوس عادت ندارید, پیش از این که کار برایتان کاملا آسان شود,

باید بر کارهایی مسلط شوید. X امکان می دهد که اعمال مجزایی – احتمالا بیشتر از آن چه فکر می کنید – را با ماوس انجام دهید .

کار با Xterm

علیرغم تمام هیاهوی موجود درباره ارزش X Window System به عنوان یک رابط گرافیکی کاربر برای یونیکس, دریافته ایم که پر استفاده ترین برنامه Xterm, X می باشد . در اصل چون ابزارهای X به اندازه کافی پیشرفته نیستند تا به طور کامل اعلان فرمان را مخفی و پوسته ها را منسوخ کنند, اغلب کاربران یونیکس هنوز هم باید فرمانهای یونیکس را در اعلان پوسته فرمان وارد کنند . Xterm رابط به X را طوری اداره می کند که تمام برنامه های قدیمی مبتنی بر متن, مانند ویرایشگر فرمان vi یا نامه رسان الکترونیکی elm, همچنین فرمانهای یونیکس, همان طوری که در شکل نشان داده می شود, به خوبی در داخل آن کار می کنند . استفاده از سیستم پنجره بندی گرافیکی فقط برای پنجره های خط فرمانی عجیب به نظر می رسد اما Xterm بیش از یک خط فرمان ساده را فراهم می آورد :

می توانید اندازه و محل پنجره Xterm , فونتها (و اندازه فونت), همچنین رنگهای پس زمینه و پیش زمینه آن را کنترل کنید .

در هر زمان می توانید بیش از یک پنجره Xterm بر روی صفحه داشته باشید و عمل بریدن و چسباندن را از بین آنها انجام دهید . این پنجره ها می توانند بر روی هم یا در کنار هم قرار بگیرند .

Xterm دارای نوار پیمایشی مفید برای بازبینی فرمانهای قبلی یا خروجی طولانی برنامه های پیچیده می باشد . در حقیقت محیط استاندارد X ما شامل دو یا سه پنجره Xterm خیلی بزرگ بر روی صفحه می باشد . این امر پایه ای را برای یک محیط پرحاصل طراحی و توسعه نرم افزار در یونیکس فراهم می آورد .

اگر صفحه نمایش متنی ۸۰ ستون در ۲۵ خط استاندارد را دوست دارید, آنگاه صفحه نمایش متنی ۸۰ ستون در ۴۶ خط را خیلی بیشتر دوست خواهید داشت, به خصوص اگر بتوانید دو تا از این صفحه های نمایش را در کنار هم داشته باشید . برپایی بهتری نسبت به صفحه های کاراکتری مجازی در بسیاری از سیستمهای ۳۸۶/۴۸۶ عرضه می گردد .

راه اندازی xterm

می توانید با فرمان زیر پنجره xterm را (معمولا از xterm دیگری) راه اندازی کنید :

```
$ xterm &
```

این فرمان, xterm را در پس زمینه شروع می کند . برای این کار باید در حال اجرای سیستم X باشید . معمولا xterm را در پس زمینه راه اندازی می کنید, به طوری که بتوانید در پایانه فعلی خود به کار ادامه دهید . ممکن است بخواهید با ویرایش پرونده .xinitrc خود پیکر بندی راه اندازی X را طوری ترتیب دهید که بیش از یک پنجره xterm – همگی در پس زمینه – به راه بیاندازید .

ویرایش متن در یونیکس

شاید ایجاد و ویرایش متن متداول ترین وظیفه مورد استفاده در کار روزمره تان باشد . هنگامی که درباره آن فکر می کنید, می بینید که اغلب نیازهای محاسباتی شما با ویرایشگرهای متن پر می شوند . نامه ها و یادداشتها با ویرایشگر نوشته می شوند . گزارشها از طریق ویرایشگر ایجاد می شوند . لیستهایی که به عنوان پایگاههای داده ای غیر رسمی عمل می کنند توسط ویرایشگر ایجاد می شوند .

یونیکس ویرایشگرهای بسیاری دارد که vi , ed **و** emacs **متداول ترین آنها می باشند . بخشی از خصوصیات این ویرایشگرها یکدیگر را می پوشانند (**vi **در واقع نگارش توسعه یافته ای از** ed **است), و تصمیم گیری برای انتخاب ویرایشگر مورد استفاده (که نیاز به استفاده معمول از یک ویرایشگر هم خواهید داشت) بستگی به در دسترس بودن و زرق و برق ویرایشگر دارد . هر ویرایشگری در هر سیستمی قابل دسترس نیست : به عنوان مثال,** vi **تنها در** SCO UNIX **وجود دارد, و** emacs **در حین این که از طریق منابع گوناگونی به طور رایگان در دسترس قرار دارد, معمولا بخشی از یک سیستم استاندارد یونیکس می باشد .**

ویرایشگر vi

Vi **به معنی ویرایشگر بصری (**visual editor **) می باشد, هنگامی که ابتدا معرفی گردید به عنوان پیشرفت بزرگی در نظر گرفته شد (در واقع هنگامی که با** ed **مقایسه می گردید,**

پیشرفت بزرگی به حساب می آمد) . تقریبا هر سیستم یونیکس با vi عرضه می گردد و همین امر آن را به یکی از در همه جا حاضرترین بخشهای نرم افزار در دنیای یونیکس تبدیل ساخته است . vi به منظور ایجاد ویرایش پرونده های اسکی استفاده می شود . vi در وضعیتهای مختلفی- ایجاد دست نوشته های پوسته و پیامهای پستی, یا ویرایش پرونده های سیستمی یونیکس, مانند پرونده های profile. و login. - کاربرد دارد .

دو راه برای راه اندازی vi - بدون بارگذاری پرونده - وجود دارد :

$ vi

یا برای راه اندازی vi و بارگذاری پرونده :

$ vi filename

که در آن filename نام پرونده ای است که ایجاد یا ویرایش می شود . اگر vi را بدون پرونده ای راه اندازی کنید, صفحه تصویری غالبا خالی, با مکان نمایی در گوشه بالایی و سمت چپ و مجموعه ای از کاراکترهای مد (~), یا پوچ در طرف سمت صفحه تصویر ظاهر میگردند . کاراکترهای پوچ به ما می گویند که هیچ چیزی بر روی صفحه نیست, چون همه کاراکترهای تایپی با نمادهای بر روی صفحه بیان نمی گردند - فضاهای خالی و بازگشت های خطی (که توسط کلید Enter یا Return تولید می شوند) ناپیدا هستند . با پرکردن صفحه از طریق تایپ, کاراکتر پوچ ناپدید می گردند .

کار با vi

کار با vi در دو حالت درج و فرمان انجام می شود . هنگامی که vi را راه اندازی می کنید, به طور خودکار در حالت فرمان قرار داده می شوید . در این حالت تمام ضربه های کلیدی به عنوان فرمان تفسیر می شوند . در این نقطه است که vi به مبتدیان پشت پا می زند : vi ورودی چندانی را از صفحه کلید نمی پذیرد, و اطلاعات دقیق اندکی در نتیجه این ضربه های کلیدی به کاربر برگردانده می شود. از آن جایی که تا کنون فرمان نداریم, کار در حالت فرمان بی ثمر است (مجموعه ای بوق, نشان دهنده پیغامهای خطا نیز تولید می کنیم) پس اولین قدم رفتن به حالت درج است . در این حالت می توانیم متن را وارد کنیم و جلوی به صدا در آمدن این بوقهای آزاردهنده را بگیریم . با تایپ I به حالت درج بروید ؛ با این کار در ابتدای صفحه قرار می گیرید . توجه کنید که vi با رسیدن به لبه های صفحه, کلمات را نمی پوشاند . اگر می خواهید به خطی پایان دهید, باید خودتان دکمه Enter (یا Return) را فشار دهید . اگر بعد از تایپ متن می خواهید که به حالت فرمان بازگردید, این کار را با فشار دادن کلید Esc انجام دهید (اگر از قرار داشتن در حالت فرمان مطمئن نیستید, چند بار کلید Esc را فشار دهید . تمام کاری که انجام می دهید, به صدا در آوردن چند بوق است و کاری بر روی پرونده فعلی انجام نمی گیرد) . vi هیچ نشانه ای در مورد حالتی که در آن قرار دارید در اختیارتان قرار نمی دهد . vi ویرایشگر متن بسیار حداقل گرایی می باشد

(اگر می خواهید در هنگامی که vi در حالت درج قرار دارد این موضوع را به شما بگوید, هنگامی که در حالت فرمان می باشید, تایپ کنید :

: set smd

این فرمان به vi می گوید که در حالت درج, عبارت INSERT MODE را در گوشه پایین و سمت راست پنجره نمایش می دهد).

در حالت فرمان می توانیم اعمال ویرایشی گوناگونی را انجام دهیم . پیش از اغلب فرمانها در حالت فرمان, دو نقطه (:) قرار می گیرد . در این موارد از توانایی جستجوی vi به منظور یافتن اولین نمونه system استفاده خواهیم کرد . خصوصیت مفید vi در توانایی جستجویش قرار دارد . در حالت فرمان /system را تایپ کنید و آنگاه Enter را فشار دهید. همان طوری که در پایین صفحه می توانید ببینید, vi در نتیجه صدور این فرمان اطلاعات حداقلی را فراهم می کند . به این باز خورد فرمان خط وضعیت گفته می شود .

نکته :

درباره فرمانهایی که در حالت فرمان وارد می شوند ناسازگاری شدیدی وجود دارد : قبل از بعضی دو نقطه (:) قرار می گیرد, قبل از بعضی دیگر ممیز (/) قرار می گیرد, و قبل از بعضی دیگر هیچ چیزی نمی آید .

ویرایشگر emacs

Emacs , ورد استار یونیکس, ویرایشگر متن معروفی می باشد که استفاده از آن نیز لزوما آسان نمی باشد . emacs با هر نگارش یونیکس عرضه نمی گردد . به عنوان مثال, emacs به عنوان بخشی از توزیع System V Release 4 عرضه نمی شود . البته فروشندگان بسیاری مناسب دیده اند که emacs را با سیستمهای یونیکس خود ارائه دهند .

اولین نگارش emacs (اگر چه برای یونیکس نبود) توسط رهبر Free Software Foundation توسط ریچارد استالمن نوشته شد . نگارشهای بسیاری از emacs , از جمله نگارشی که توسط Free Software Foundation توزیع می شود در اطراف پراکنده می باشند

راه اندازی emacs

دو راه برای راه اندازی emacs در خط فرمان وجود دارد :

$ emacs

اگر می خواهید با راه اندازی emacs پرونده ای بار گذاری شود, نام پرونده را به نام آرگومانی در خط فرمان قرار دهید :

$ emacs filename

ممکن است صفحه شروعی حاوی مقدمه ای کوتاه در مورد emacs و پیغام کمکی نمایش داده شود .

برای اغلب کاربران صفحه ای غالبا خالی به همراه خط وضعیتی در پایین صفحه نمایش داده می شود . این خط وضعیت نام پرونده ای را که ویرایش می شود به همراه اطلاعاتی درباره اندازه پرونده بر حسب خط و کاراکتر, همچنین موقعیت شما در پرونده نمایش می دهد .

Emacs بر خلاف vi یا ed فقط در یک حالت کار می کند, بنابراین نیازی نیست که درباره فشردن کلید Esc برای ورود به حالت فرمان نگران باشید . بعد از راه اندازی emacs به طور مستقیم می توانید متن را تایپ کنید . Emacs برای جا دادن متن در داخل حدود صفحه نمایش آن را نمی پوشاند (به طور پیش گزیده, اگر چه این وضعیت قابل تغییر است), بنابراین در انتهای هر خط باید کلید Enter یا Return را فشار دهید .

سیستم کمک در emacs

Emacs با کمال تعجب دارای سیستم ابتدایی کمک می باشد که ممکن است به دردتان بخورد (چون خصوصیات کمک ـ به غیر از صفحات man ـ در دنیای یونیکس به شدت نادر می باشند, وجود چنین سیستمی در emacs, به منظور احضار سیستم کمکی یکی از فرمانهای زیر را تایپ خواهید کرد :

Ctrl-H

F1

Esc-?

Meta-?

Meta-x

متاسفانه نگارشهای متفاوت emacs به شکل متفاوتی با سیستم کمک برخورد می کنند, با این وجود Ctrl-H وسیله برای وارد شدن به سیستم کمک می باشد . به منظور به دست آوردن سرفصلی از تمام موضوعات سیستم کمک, استفاده از فرمان زیر مناسب است :

Meta-x a

علاوه بر این فرمانها ,می توانید کارهای بسیار دیگری را نیز در emacs انجام دهید . از آنجایی که emacs شامل یک مفسر تهیه شده Lisp می باشد , قادرید با نوشتن برنامه های کوچک Lisp , emacs را مطابق میل خودتان اختصاصی کنید . بسیاری از افراد emacs را طوری پیشرفته ساخته اند که از آن می توانید به عنوان پوسته , وسیله ای برای خواندن پست الکترونیکی و اخبار اینترنت /یوزنت خود , و انجام بازی go , همگی در داخل یک رابط (یک رابط برای فرمانروایی بر آنها و گرفتار نمودنشان در تاریکی) استفاده کنید .

یونیکس در شبکه

در شبکه توانایی اتصال کامپیوترهای متفاوت , همیشه بخش مهمی از سیستم عامل یونیکس بوده است . سیستم های یونیکس توانایی فراخوانی مستقیم سایر سیستم های یونیکس را دارند , و هزاران سیستم یونیکس (همچنین غیر یونیکس) در شبکه هایی با مدیریت ضعف به یکدیگر مشخص شده اند که در این شبکه ها , پرونده ها , اخبار و برنامه ها را در سرتاسر جهان توزیع می کنند . علاوه بر این , اغلب شرکتهایی که دارای ایستگاه های کاری یونیکس می باشند , با به کار بردن قراردادهای استفاده از شبکه, مانند اترنت یا Token

Ring , کامپیوترهای خود را توسط اتصالهای شبکه ای سریع , با قدرت به یکدیگر متصل کرده اند. قبلاً بخش کلیدی اتصال پذیری داخل یونیکس , یعنی پست الکترونیکی را بررسی کرده ایم . این فصل بحث را با توضیح ابزارهای یونیکس برای اتصال به سیستمهای راه دور گسترش می دهد . فصل بعدی یوزنت , اینترنت و World Wide Web را پوشش خواهد داد .

ارتباطات غیر رسمی توسط فرمانهای UUCP

یونیکس می تواند از روشهای متفاوت بسیاری برای انتقال اطلاعات بین سیستم ها استفاده کند . در ابتدا uucp (Unix-to-Unix Copy Program) برای برقراری ارتباط بین سیستم ها از طریق خطوط معمولی تلفن , نوشته شد . برنامه uucp امکان می دهد که پرونده را از یک سیستم به سیستم دیگر کپی کنید . امروزه این اتصالها بین همان خطوط تلفن از طریق مودم (در تمام سرعتها , از ۲۴۰۰ بیت در ثانیه تا ۱۹/۲ کیلوبایت در ثانیه) , سیم بندی مستقیم , شبکه ای محلی , یا شبکه ای گسترده که از طریق خطوط تلفن اختصاصی متصل می باشد , صورت می گیرند . با وجود این کار سازوکارهای اتصال تغییر کرده ,اما سیستم اولیه UUCP بدون تغییر و مستقل از سازوکار باقی مانده است که همین موضوع زندگی را برایتان ساده تر می سازد . به عنوان یک کاربر نیازی نیست که از خصوصیات سازوکار اتصال آگاه باشید ؛ تمام آن چیزی که باید بدانید چگونگی دستیابی به برنامه های کمکی برقراری ارتباط می باشد .

نکته :

سروکار داشت با uucp و برنامه های کمکی استفاده از شبکه در سطح پیکربندی , مسئولیت پیشرفته ای است که به عهده مدیران سیستم که مناسب انجام وظایف پیچیده مرتبط با ماشینهای شبکه ای یونیکس باشند , گذاشته می شود .

برای سرپرستی اتصالهای یونیکس به دنیای خارج برنامه ای بسیار بالاتر وجود ندارد . برنامه های کمکی ارتباطات خیلی شبیه به هر چیز دیگری در دنیای یونیکس به تنهایی کاملاً کوچک هستند و به درد مصارف محدودی می خورند ؛ تنها هنگامی که این برنامه ها در کنار هم قرار می گیرند می توانند سیستم ارتباطاتی قدرتمندی را به وجود آورند. چرا به دنیای خارج متصل شویم ؟ بعضی از شرکتها , دفاتر کار خود را که در مسافتهای دور گسترده شده اند از طریق خطوط اختصاصی تلفن به یکدیگر متصل می کنند تا ارتباطات آنی بین کارمندان تضمین شود .سایر شرکتها توسط مودم از راه خطوط تلفن به شبکه UUCP متصل می شوند . این شبکه مجموعه ای از کامپیوترهای یونیکس می باشد که پست الکترونیکی و پرونده ها را در سرتاسر جهان در امتداد خود عبور می دهند. UUCP در وضعیتی گیج کننده تر به فرمانی خاص (uucp) و مجموعه ای از فرمانهای مرتبط (که بیشتر آنها با uu شروع می شوند) اشاره دارد . در این قسمت منظور از uucp , فرمان خاص uucp خواهد بود , در حالی که UUCP به مجموعه کلی فرمان اشاره خواهد داشت .

مبادی اولیه UUCP

فرمانهای UUCP به ماشینها امکان می دهند که از طریق اتصال های شبکه ای یا تلفنی با یکدیگر ارتباط برقرار کنند . دامنه فعالیت این فرمانها محدود است و برای مقاصد ابتدایی ارسال پرونده ها , پست الکترونیکی , و (بعضی اوقات) اخبار یوزنت آماده کار می گردند . در این قسمت به جای پوشش دادن به تمام فرمانهای uucp فقط فرمانهایی با کاربرد بیشتر را توضیح خواهم داد .

قبل از به کار بردن uucp , باید از ماشینهای متصل به ماشین خودتان آگاه شوید . فرمان uuname این کار را انجام می دهد :

```
$ uuname
geisha
spike
khan
kirk
picard
```

چرا این اطلاعات اهمیت دارد ؟ چون در فرمان uucp باید اسامی ماشینها را مشخص کنید .

استفاده از UUCP

در واقع , فرمان uucp به منظور کپی کردن پرونده ها از یک ماشین به ماشین دیگر استفاده می شود . در نگاه اول ممکن است فکر کنید که در این مرحله از رشد اینترنت و شاهراههای اطلاعاتی این تکنولوژی به شکلی باور نکردنی مرتجع است و به لحاظ مفهومی نیز اینطور می

باشد. اما اگر بخواهیم واقع بین باشیم باید بگوییم در دنیایی که کامپیوترهای متصل به یکدیگر بسیار متدوال می باشند , فرمان uucp دارای گسترده ترین کاربرد خود می باشد . در این وضعیت ممکن است که سیستمهای متحد با دنیای خارج ارتباطی نداشته باشند , بلکه فقط به سایر سیستم های متحد متصل شوند . بهترین راه برای فهمیدن فرمان uucp این است که آن را به عنوان نگارش توسعه یافته فرمان cp در نظر بگیرید . به جای اینکه پرونده ها را از فهرست محلی منتقل و به آن ارسال کنید , با استفاده از فرمان uucp پرونده را از ماشین دیگری منتقل و به آن ارسال می نمایید .

فرمان uucp دارای فایده دیگری نیز می باشد : با در نظر گرفتن همه جوانب , این فرمان هنگامی که همه چیز به درستی برپا شده باشد , روش امن تری برای منتقل ساختن پرونده ها می باشد . فرمان uucp همچنین برنامه های کمکی uucp می توانند اصول بسیار خاصی را درباره محل مورد نظر برای منتقل ساختن پرونده ها از آن یا ارسال پرونده ها به آن تعیین نمایند .

بیایید به یک خط فرمان uucp شاخص نگاهی بیاندازیم :

```
$ uucp chap9.txt spike!/user/spool/uucpublic/chap9.txt
```

ممکن است که این خط فرمان طولانی و غیر معمولی به نظر برسد , اما نگاهی دقیق تر به آن نشان خواهد داد که بسیار ساده می باشد .

بخش uucp **فرمان به فرمان** uucp **اشاره دارد . به دنبال این بخش نام پرونده ای که کپی می شود (** chap9.txt **) قرار می گیرد .**

بخش بعدی برای کاربران تازه کار , گیج کننده ترین بخش می باشد . این قسمت از فرمان , مقصد پرونده را با نام ماشین و فهرست بیان می کند . در این مورد , spike **به نام ماشین اشاره دارد ؛** uucp **با توجه به علامت تعجبی (!) که در انتهای نام ماشین می آید از این موضوع آگاه می شود . بعد از علامت تعجب فوراً فهرست مقصد قرار می گیرد . فهرست** /user/spoll/uucppublic **مقصد متداولی برای اغلب سایتهای** UUCP **می باشد .**

شکل کلی فرمان uucp **به صورت زیر است :**

$ uucp sourcefile destinationfile

نکته :

چون علامت تعجب در پوسته **C** بیانگر تاریخچه فرمان است , پوسته بر روی این علامت متوقف خواهد شد . برای این که به پوسته **C** بگویید که علامت تعجب بخشی از فرمان **uucp** می باشد , ممیز وارونه ای (\) را قبل از آن قرار دهید :

% uucp chap9.txt spike\!/user/spool/uucppublic/chap9.txt

اگر دارای مجوز های مناسب باشید , می توانید فرمان uucp **را به منظور گرفتن پرونده ها از ماشین دیگری به کار ببرید . در این مورد خط فرمان** uucp **مثال قبل را تغییر خواهید داد .اصول یکسان می باشند : از فرمان** uucp **برای فهرست نمودن پرونده منبع و آنگاه**

مقصدش استفاده کنید . در این حالت پرونده منبع , پرونده راه دور , و فهرستی محلی مقصد انتقال می باشد :

$ uucp spike!/user/spool/uucppublic/chap9.txt

/user/spool/uucppublic

نکته :

هر پرونده در یونیکس دارای سه سطح دسترسی برای سه نوع کاربر می باشد . هنگامیکه فرمان LS-1 را برای لیست کردن پرونده ها اجرا می کنید , ۹ کاراکتر آخر از اولین ستون که شامل ۱۰ کاراکتر می باشد , مشخص کننده مجوزهای هر پرونده می باشد . (کاراکتر اول نوع پرونده را مشخص می کند) این مجوزها از سه کاراکتر r , نماد مجوز خواندن , w نماد مجوز نوشتن و x نماد مجوز اجرا , تشکیل شده اند . سه مجوز اول از سمت چپ , برای مالک پرونده , سه مجوز بعدی برای یک گروه کاربران خاص و سه مجوز آخر , برای سایر کاربران می باشد .

در هنگامی که توسط فرمان uucp **پرونده ای را به کامپیوتر دیگری منتقل کنید ؛ فقط لازم است که فهرست مقصد را مشخص کنید ؛ فرمان فرض می کند که نام پرونده همان باقی خواهد ماند و فهرست مقصد بر روی ماشین محلی موجود است .**

مشکلات بالقوه با فرمان uucp

فرمان uucp **, همچنین مجموعه فرمان بزرگ تر** UUCP **محفوظ از خطا نمی باشد . شاید بزرگترین ناامیدی این باشد که فرمان** uucp **محاوره ای نیست و هیچ راهی برای نظارت بر**

وضعیت انتقال پرونده وجود ندارد . می توانید به uucp بگویید که بعد از تکمیل انتقال پرونده , رسیدی را از طریق پست الکترونیکی بفرستد :

$ uucp –m chap9.txt spike!/usr/spool/uucppublic/chap9.txt

اما اگر پست الکترونیکی تصدیق کننده را دریافت نمی کنید , آنگاه فرض کنید که انتقال با مشکل مواجه شده است . ولی یافتن این امر آسان نیست . uucp به عنوان بخشی از فرایند انتقال پرونده ثبتی را نگهداری می کند و با وجود این پرونده یافتن علت عدم موفقیت انتقال ساده تر می گردد . برای دست یافتن به این پرونده از فرمان uulog استفاده کنید :

$ uulog

خروجی این فرمان بسیار بزرگ است و احتمالاً به منظور کشف و رفع عیب بسیار زیاد می باشد . بنابراین بهتر است که uulog را با نام ماشینی ترکیب کند :

$ uulog –sspike

آنگاه باید اطلاعات خروجی را بخوانید و استنباطی از آنها انجام دهید . انتقالهای موفق پرونده با REMOTE REQUESTED یا OK پایان می یابد .

نرم افزار رایگان و ftp

بسته به نگارش یونیکس مورد استفاده , ممکن است دارای مجموعه ای از برنامه های کمکی باشید که امکان بدهند با استفاده از قراردادهای TCP/IP شبکه به طور مستقیم به کامپیوتر دیگری متصل شوید . BSD و System V Release 4 هر دو از این قراردادها استفاده می کنند و بعضی از تولیدکنندگان در حالی که هنوز از نگارشهای قدیمی تر یونیکس را

پشتیبانی می نمایند این قراردادها را پیاده سازی کرده اند. این قراردادها به کامپیوترهای یونیکس و غیر یونیکس امکان می دهند که به شکل موثری با یکدیگر ارتباط برقرار کنند , و به شبکه ها امکان می دهند که به یکدیگر متصل شوند. فرمان ftp مفیدترین کاربرد TCP/IP می باشد . این فرمان امکان می دهد که پرونده ها را از ماشینهای راه دور (ماشینهایی که کامپیوترتان با آنها در شبکه قرار دارد) بگیرید . به خاطر تاریخچه یونیکس در دایره عمومی , تعداد زیادی نرم افزار رایگان در دسترس قرار دارند . به عنوان مثال در هر ماشینی که امکان منتقل ساختن پرونده ها از آن وجود داشته باشد , emacs در دسترس قرار دارد . یا می توانید مجموعه کامل نرم افزاری Free Software Foundation (FSF) به نام GNU را به دست آورید . این مجموعه نرم افزاری شامل GNU Emacs (ریچارد استالمن رئیس FSF می باشد و نگارش اولیه emacs را نوشته است) , GCC (کامپلر C با استفاده از گسترده) , GNU Chess (که به کامپیوترهای متفاوت بسیاری برده شده است ؛ حتی دیده ایم که برای استفاده توسط کامپیوترهای شخصی در حال اجرای ویندوز مایکروسافت کامپایل می شود !) و برنامه های دیگر بسیاری در شکل کد منبع می باشد . به طور مشابهی می توانید کل X Window System را از طریق ftp بی نام به دست آورید . البته حجم چنین انتقال داده ای زیاد می باشد .

نکته :

توانایی گرفتن پرونده ها از طریق ftp در نظر اندازه های وب اینترنت , مانند NCSA Mosaic تعبیه می شود .

استفاده از ftp

فرمان ftp می تواند برای اتصال به سایر کامپیوترهای شبکه که ftp را اجرا می کنند , استفاده شود . اگر کامپیوترتان به اینترنت متصل باشد , با استفاده از ftp می توانید به پرونده های سایر کامپیوترهای اینترنت در سطح جهان دستیابی داشته باشید . کامپیوترهایی که با آنها در شبکه می باشید , ممکن است سیستم عامل یونیکس را اجرا کنند و یا در حال اجرای سیستم عامل دیگری باشند ؛ این استقلال از سیستم عامل کاربرد ftp را بسیار گسترده می سازد. ftp نرم افزاری محاوره ای می باشد , یعنی در زمانهای خاصی اطلاعاتی را از شما درخواست می کنند. ftp نرم افزاری محاوره ای می باشد , یعنی در زمانهای خاصی اطلاعاتی را از شما درخواست می کند . ftp را با فرمان زیر راه اندازی کنید :

$ ftp

ftp>

اعلان ftp نمایش خواهد یافت . در این اعلان فرمانهای خاص ftp را وارد می کنید . به منظور به دست آوردن فهرستی از فرمانهای در دسترس , علامت سوال (؟) یا help را در پای اعلان تایپ نمایید :

ftp>?

یا

ftp>help

متداولترین فرمانهای ftp:

فرمان	معنی
ascii	از آسکی به عنوان نوع انتقال پرونده استفاده می کند .
bell	با تکمیل انتقال پرونده زنگ را به صدا در می آورد .
binary	از دودویی به عنوان نوع انتقال پرونده استفاده می کند .
quit یا **bye**	به جلسه **ftp** خاتمه می دهد .
cd	بر روی ماشین راه دور فهرست جاری را تغییر می دهد .
close	به اتصال **ftp** با کامپیوتر راه دور پایان می دهد , اما برنامه **ftp** محلی را در حال اجرا نگه می دارد .
delete filename	بر روی ماشین راه دور پرونده **filename** را حذف میکند .
get filename	پرونده **filename** را از ماشین راه دور می گیرد .
get filename filename2	پرونده **filename** را از ماشین راه دور می گیرد و آن را با عنوان **filename2** به شکل محلی ذخیره می کند .
help	فرمانهای در دسترس را فهرست می کند .
mput filename	پرونده محلی **filename** را به ماشین راه دور کپی می کند
pwd	فهرست جاری بر روی ماشین راه دور را نشان می دهد .

توسط ftp به سادگی می توان پرونده ها را از ماشین راه دوری منتقل ساخت . در نظر بگیرید که می خواهیم پرونده هایی را از ماشینی به نام mn.Kevin.com (این ماشین واقعی نیست) بر روی اینترنت بگیریم . با فرض این که ماشین فوق ftp بی نام را پشتیبانی می کند و البته ماشین فرضی ما نیز چنین امکانی را فراهم می آورد , فقط نام ماشین را بر روی خط فرمان مشخص کنید:

$ ftp mn.Kevin.com

اگر اتصال موفق باشد , پیام تاییدی به همراه اعلان برقراری ارتباط دریافت خواهید کرد . از آنجایی که عمل ftp بی نام می باشد , از anonymous به عنوان نام برقراری ارتباط استفاده کنید :

Name :anonymous

آنگاه کلمه عبور از شما درخواست می گردد . اغلب سیستمها مجبورتان می کنند که نشانی پست الکترونیکی خود را ارائه کنید , در حالی که بعضی دیگر به عنوان کلمه عبور احتیاج به کلمه guest دارند . سپس اعلان ftp نمایش داده خواهد شد .

سیستم راه دور طوری برپا شده است که امکان دستیابی محدودی در اختیارتان قرار می دهد . یعنی قابلیت مانورتان بسیار محدود می شود , و پرونده های مورد نظرتان معمولاً در مکان نزدیکی قرار دارند . می توانید با فرمان cd یونیکس به فهرست های دیگری بروید .

پیش از شروع به جستجوهای بزرگ به دنبال پرونده ها, بایستی مطالبی را درباره پرونده هایی که منتقل می کنید , بدانید . اگر این پرونده ها , پرونده های ساده C در شکل غیر

فشرده و اسکی می باشند , می توانید با استفاده از تنظیمهای پیش گزیده ای انتقال پرونده آنها را منتقل کنید . اغلب پرونده های بزرگ تر , بخصوص پرونده های دودویی , به شکل فشرده ذخیره می شوند بنابراین زمان انتقال کمتری می برند . این پرونده های فشرده با tgz, .z,.Z **پایان می یابند , به طوری که فوراً قابل تشخیص هستند . به منظور ساختن پرونده های فشرده از آنجایی که در حالت منتقل ساختن پرنده های دودویی می باشید , باید به حالت دودویی بروید . این کار با فرمان زیر انجام دهید :**

ftp>binary

هنگامی که در فهرست جاری حاوی پرونده , هیچ اعلانی بر روی سیستم وجود نخواهد داشت و قادر به وارد نمودن هیچ ضربه کلیدی نمی باشید . بعد از انتقال موفق پرونده , پیغامی شبیه به این را خواهید دید :

Transfer complete

بعد از این که نیاز خود به پرونده های مورد نظر را برطرف نمودید , با فرمان bye **به اتصال خاتمه دهید :**

ftp>bye

با پرونده چه کار کنم ؟

اگر پرونده ای اسکی را منتقل نموده اید , می توانید با استفاده از هر ویرایشگری , از جمله vi **یا** emacs **آن را ببینید . اگر پرونده فوق کد منبع می باشد , می توانید آن را برای استفاده در سیستم خود کامپایل کنید ؛**

اگر پرونده ای دودویی و فشرده را منتقل ساخته اید , باید با استفاده از tar, unpack, uncompress **یا** gzip **آن را در خط فرمان از حالت فشرده (یا شاید از حالت بایگانی) خارج کنید . چگونه می فهمید که از کدام یک استفاده نمایید ؟**

اگر پرونده با Z **.خاتمه یابد , از فرمان** uncompress **استفاده کنید .**

$ uncompress filename.Z

اگر پرونده با z. **خاتمه می یابد , از فرمان** unpack **استفاده کنید ؛**

$ unpack filename .z

اگر پرونده با gz. **یا** tgz. **خاتمه یابد , از فرمان** gunzip **استفاده کنید :**

$ gunzip filename.gz

اگر بعد از به کار بردن این فرمانها پرونده دارای پسوند tar **می باشد , برای خارج ساختن آن از حالت بایگانی شده , از فرمان** tar **استفاده کنید :**

$ tar xvf filename.tar

این موارد در کل قوانین خوبی را به منظور پیروی تشکیل می دهند . اما در هنگام کار با پرونده های فشرده چند نکته را باید به خاطر سپرد .

اول از همه باید گفت که فرمان guzip **, همراه فرمان** gzip **, بخشی از مجموعه استاندارد یونیکس نمی باشد , این فرمان توسط** Free Software Foundation **ایجاد و نگهداری گردید. بسیاری از نگارشهای رایگان لینوکس , مانند لینوکس ,** gzip **را همراه ندارند . اگر در سیستم خود** agzip **ندارید , مدیر سیستم را برای تهیه آن تشویق کنید .** gzip **به طور رایگان در اینترنت , کتابها و** CD-ROM **(های) حاوی نرم افزار رایگان یونیکس در دسترس قرار دارد .**

چرا gunzip **این قدر مفید است ؟ چون هر پرونده فشرده شده یونیکس , حتی آنهایی را که با فرمانهای** comress **و** pack **فشرده شده اند را از فشردگی خارج خواهد کرد . برعکس ,** unpack **پرونده های فشرده شده توسط فرمان** compress **را از حالت فشرده خارج نخواهد کرد , و فرمان** uncompress **نیز پرونده های فشرده شده توسط فرمان** unpack **را از حالت فشرده را از حالت فشرده خارج نخواهد شد .**

برای استفاده از gzip **لزومی به وجود فرمان** gunzip **در سیستم خود نمی باشد . خط فرمان :**

$ gzip -d file.gz

کاری مشابه با خط فرمان زیر انجام می دهد :

$ guzip file.gz

دوم اینکه در هنگام استفاده از فرمانهای uncompress **,** unpack **, یا** guzip **, پرونده ای که از حالت فشرده خارج می شود جایگزین پرونده فشرده شده اولیه خواهد شد ؛ به عنوان مثال ,** filename **جایگزین** filename **می شود . اگر نمی خواهید که پرونده فشرده شده**

اولیه را از دست بدهید , نسخه ای از پرونده filename.gz **را در فهرست دیگری ایجاد کنید.**

استفاده از فرمان rlogin

فرمان rlogin **امکان می دهد که از راه دور با کامپیوتر دیگری در شبکه خود ارتباط برقرار کنید . البته بر روی ماشینی که قصد برقراری ارتباط با آن را دارید , باید حساب کاربری معتبری داشته باشید . برای برقراری ارتباط با ماشین باید نام آن را بدانید . به منظور برقراری ارتباط با آن را دارید , باید حساب کاربری معتبری داشته باشید . برای برقراری ارتباط با ماشین باید نام آن را بدانید . به منظور برقراری ارتباط با ماشینی به نام** nicollet **, از فرمانی شبیه به این استفاده می کنید :**

```
$ rlogin nicollet
password :
```

در اعلان password **احتمالاُ باید کلمه عبور خود در ماشین** nicollet **را وارد کنید . این کلمه عبور می تواند با کلمه عبور مورد استفاده در ماشین جاری یکسان یا با آن متفاوت باشد . بعد از برقراری ارتباط مشغول به انجام محاسبه بر روی ماشین راه دور می وشید و می توانید هر فرمان یونیکس را وارد کنید . مانند روش معمول قطع ارتباط امکان قطع ارتباط با ماشین راه دور وجود دارد :**

```
$ logout
connection closed.
```

$

نکته :

بعد از قطع ارتباط با ماشین راه دور , به اعلان فرمان در ماشین اولیه خود بازمیگردید .

شکل اولیه rlogin **بدین صورت است :**

$ rlogin hostname

که در آن hostname **نام ماشین مورد نظر برای برقراری ارتباط می باشد .**

استفاده از فرمان telnet

فرمان telnet **بسیار شبیه به** relogin **کار می کند و امکان می دهد که به طور مستقیم به ماشین راه دور متصل شوید .این فرمان به عنوان بخشی از جعبه ابزار مورد استفاده در اینترنت در نظر گرفته میشود و به همین دلیل در طی سالها به محبوبیت زیادی دست پیدا کرده است .**

telnet **امکان می دهد که فرمانی را به طور مستقیم برروی ماشین راه دوری اجراکنید ودر همین حال نتایج را بدون هیچ کمکی نمایش دهید ,یا فرمان خاصی را بر روی ماشین راه دوری اجرا کنید .**

در telnet **تنها باید از نشانی ماشینی که به آن متصل می شوید ,مانند** sunite .unc.edu , **آگاه باشید . نحوه کار در زیر به تصویر کشیده می شود .**

$ share

telnet>open sunsite.unc.edu

sunsite.unc.edu **خدمات گوناگونی را به عموم ارائه می کند که در این مورد خدمات فوق در حول پایگاههای داده ای** WAIS **متمرکز است . سایر سایتهای** telnet **ممکن است که چنین گوناگونی گسترده از خدمات را ارائه نکنند ؛ سایتی مانند** archie.Rutgers.edu **تنها جستجو های** archive **را عرضه می دارد .**

sunsite.unc.edu **سایت عمومی اینترنت است . اگر در داخل سازمان تجاری خود از** telnet **استفاده کنید , قوانین کمی متفاوت خواهند بود . در اینجا ,** sunsite.unc.edu **دستیابی عمومی را عرضه می دارد ؛ نیازی به داشتن حساب و کلمه عبور در** sunsite.unc.edu **ندارید . اما** sunsite.unc.edu **در رابطه با اعمال قابل انجام توسط ملاقات کننده محدودیت قائل می شود ؛ به عنوان مثال , نمی توانید از مجموعه فرمان استاندارد یونیکس استفاده کنید , و انتخابهای شما به انتخابهای برقراری ارتباط محدود می شود . در سیستمی خصوصی به منظور برقراری ارتباط نیاز به حسابی بر روی سیستم راه دور دارید , ممکن است که تحت کنترل همان نوع محدودیتها قرار بگیرید . به یاد داشته باشید :امنیت .**

RFS و NFS

در هنگام استفاده از ftp یا relogin , بعد از فراهم آوردن نام سیستم راه دور با آن ارتباط برقرار می کنید . سیستم عامل یونیکس بسیاری دارد که به ماشینهای یونیکس امکان می دهند تا طوری متصل شوند که در نهایت ماشینهای راه دور , محلی به نظر آیند .

نتیجه نهایی , سیستم توزیع شده پرونده نامیده می شود . در این سیستم پرونده با وجود این که ممکن است ماشینهای شبکه مایلها دور از یکدیگر باشند , اما تمام این ماشینها امکان می یابند که مانند یک کامپیوتر بزرگ واقع در محلی فیزیکی عمل کنند . برای کاربر , منابع واقع بر روی ماشین راه دور محلی به نظر می رسند ؛ برای دستیابی به ماشین های راه دور احتیاج به هیچ تلاش آگاهانه ای از جانب کاربر نمی باشد .

یونیکس این اتصال را از طریق سه ابزار انجام می دهد : اشتراک پرونده از راه دور (RFS) که در ابتدا توسط AT&T طراحی شد ؛ سیستم پرونده شبکه (NFS) برای استفاده توسط شبکه و ماشینهایی که در حال اجرای سیستمهای عاملی به غیر از یونیکس می باشند , طراحی شد .

نصب و مدیریت سیستم توزیع شده پرونده کار مدیر سیستم , یا دقیقتر بگویم معمولاً چندین مدیر سیستم می باشد . نکته نهفته در سیستم توزیع شده پرونده این است که کل سیستم به طور کامل در نظر کاربر نهایی شفاف (ناپیدا) به نظر برسد . اما به لحاظ اهمیت

سیستمهای توزیع شده پرونده در دنیای یونیکس , آگاهی درباره بعضی از مفاهیم DFS یا سایر ابزارهای مورد استفاده در سیستم بخصوص شما اهمیت دارد .

سرویس دهنده ها و سرویس گیرنده ها

در هنگام بحث درباره تقریباً هر نوع شبکه سازی توزیع شده (بجز X Window System) , برای فرق گذاشتن بین ماشینهای متفاوت در شبکه , بحث سرویس گیرنده ها و سرویس دهنده ها را به میان می آوریم . کامپیوتری که پرونده ها و خدمات را برای سایر کامپیوترها فراهم می آورد , سرویس دهنده , و کامپیوتری که به این پرونده و خدمات دستیابی پیدا می کند , سرویس گیرنده نامیده می شود . چون ماشین یونیکس چند وظیفه ای می باشد و در هر لحظه توانایی انجام بیش از یک وظیفه را دارد , می تواند به طور همزمان سرویس دهنده و سرویس گیرنده باشد .

بعد از این سرویس دهنده ای سیستم پرونده خودش را به منظور استفاده توسط سرویس گیرنده ها اعلان نمود , سرویس گیرنده سیستم پرونده را برای استفاده خودش سوار کند. به عنوان مثال , ماشین " الف " ممکن است بخواهد به پرونده هایی درباره وضعیت مالی شرکت در ماشین "ب" دستیابی پیدا کند و این پرونده ها در فهرست /user/data/finances ماشین "ب" قرار داشته باشند . برای سوار نمودن پرونده ها باید فهرستی بر روی ماشین "الف" مشخص شود . پرونده های /user/data/finances در ماشین "الف" مانند پرونده های

ذخیره شده بر روی سیستم دیسک سخت ماشین "الف" ظاهر خواهند شد. مفهوم سیستم های توزیع شده پرونده بدون احتیاج به تغییر عمده به شکل زیبایی در فلسفه پیمانه ای بودن و گسترش پذیری یونیکس جای می گیرد . در طراحی و توسعه RFS , اقتصاد نقش مهمی را بازی نمود . تحت RFS فضای ذخیره سازی اضافی بدون نیاز به نصب کامپیوترهای جدید و گران می توانست به سیستم یونیکس اضافه شود ؛ هنگامی که هزینه سخت افزار نسبت به امروزه بسیار بیشتر بود , این موضوع در بخشهای داده پردازی با بودجه محدود در درجه اول اهمیت قرار داشت . به شکل مشابهی ابزارهای جانبی گران قیمت مانند مودمها و چاپگرها حداقل تحت RFS ـ می توانستند در میان سیستم های متفاوت به اشتراک گذاشته شوند.

برای تعیین منابع به اشتراک گذاشته در سیستم از فرمان share به صورت زیر استفاده کنید:

```
$ share
```

برنامه های پستی یونیکس

توانایی ارسال پیامهای الکترونیکی به افراد , گروههایی از مردم , یا هر فردی در شرکت یکی از درخشان ترین خصوصیات سیستم عامل یونیکس نمی باشد , اما به طور قطع یکی از پرکاربردترین آنهاست . سایر سیستم های شبکه سازی , بخصوص سیستم های موجود در

دنیا MS-DOS (مانند نت ور ناول) فاقد تواناییهای اولیه پست الکترونیکی (یا e-mail) می باشند , در حالی که سایر سیستمهای عاملی که سایر سیستمهای عاملی که پست الکترونیکی تعبیه شده را عرضه می دارند , فاقد دیگر تواناییهای گسترده موجود در یونیکس هستند .

این بخش به قابلیتهای پست الکترونیکی یونیکس می پردازد . در اینجا چگونگی خواندن , فرستادن , و حذف پیامهای پست الکترونیکی را توضیح داده و همچنین بعضی از نامه خوانها معروف را بررسی می کنیم .

برنامه mail تقریباً از ابتدا بخش مهمی از یونیکس بوده است . با رشد یونیکس , mail نیز تا حدی رشد یافته است . سازوکارهای واقعی پست الکترونیکی شبیه به سازوکارهای اصلی mail می باشند , تغییرات در اصل متوجه چگونگی فعل و انفعالات کاربر با برنامه mail است .

نکته :

با انفجار در علاقه به پست الکترونیکی , بسته های نرم افزاری پست الکترونیکی بسیار در دنیای یونیکس قابل دسترسی شده اند . بعضی از این بسته های نرم افزاری , مانند فرمان mail و برنامه های کاربردی xmh به همراه یونیکس و X Window System عرضه می گردند . بعضی دیگر , مانند pine,elm و mush نرم افزار رایگان می باشند و از طریق اینترنت و CD-ROM به شکل کد منبع در دسترس قرار دارند . بعضی دیگر , مانند Z-Mail تجاری هستند .

دریافت نامه

در بخش هشتم گفته شد که در هنگام برقراری ارتباط با سیستم , یونیکس شما را از نامه های وارد شده آگاه می کند . پیغامی شبیه به این را خواهید دید :

You have mail.

تا زمانی که نامه خود را در این نقطه نخوانید , این پیغام به طور مرتب مجدداً ظاهر خواهد شد , زیرا پوسته به طور خودکار طوری تنظیم شده که نامه خوانده نشده را به یادتان آورد :

برای دیدن این نامه , تایپ کنید :

$ mail

پوسته مهندسی با فهرستی از پیامهای پستی را که به ترتیب ورودشان ـ جدیدترین نامه در ابتدا ـ فهرست شده اند , پاسخ می دهد (این پیامها معمولاً در پرونده /user/mail/yourname , /user/spool/mail/yourname , یا /user/spool/mqueue/yourname قرار می گیرند). فیلد اول فرستنده پیام , و فیلدهای دوم تا پنجم زمان و تاریخ دریافت پیام , فیلد ششم تعداد خطوط پیام و اندازه آن (بر حسب بایت), و آخرین فیلد موضوع پیام را نشان می دهد.

به منظور خواندن اولین پیام در لیست , کلید Enter را فشار دهید . اگر پیام طولانی باشد , کل پیام طولانی باشد , کل پیام به شکل طوماری نمایش داده پیدا می کند . اگر بخواهید

حرکت طوماری پیام را متوقف کنید , ctrl+s و برای راه اندازی مجدد حرکت طوماری پیام ctrl+Q را تایپ کنید .

دو نوع برنامه منتظر ما می باشند : پیامی از kreichard@mcimail.com و پیامی از etc. نامه الکترونیکی از دو منبع می تواند وارد شود : سیستم خودتان و از سایر سیستمها . نامه ای که از سایر سیستمها بر روی اینترنت یا یوزنت فرستاده می شود , دارای طرح نشانی دهی منحصر به فرد خودش می باشد . نامه ای که از سیستم خودتان می آید از همان اسامی برقراری ارتباط استفاده می کند ؛ این اسامی در پرونده /etc/passwd وجود دارند .

اینترنت

اینترنت ، گسترده ترین شبکه ارتباطات کامپیوتری در جهان می باشد . بسیاری از شرکتها ، دانشگاه ها و موسسات تحقیقاتی از طریق اینترنت اقدام به ارسال و دریافت نامه می کنند . تمام کامپیوترهای موجود بر روی اینترنت سیستم عامل یونیکس را اجرا نمی کنند , اما اغلب دو طرح نشانی دهی اولیه را به اشتراک می گذارند .

طرح اول و قدیمی تر , مسیر دقیق نامیده می شود که ابتدا در یوزنت که بخشی از اینترنت را تشکیل می دهد به شهرت رسید . اصولاً در این طرح مسیر دقیق ارسال نامه خود را به سیستم پستی می گویید . این کار می تواند دردسر بزرگی باشد , بخصوص اگر که پیام پیش از تحویل به مقصد مجبور باشد از تعداد زیادی ماشین عبور کند . خوشبختانه این نشانی دهی دستی در پست الکترونیکی در حال کوچکتر و ضعیف تر شدن می باشد و به همین

دلیل نشانی های پست الکترونیکی کمتری را می بینید که با علایم تعجب نقطه گذاری شده باشند .

هنوز هم هنگامی که بحث پست الکترونیکی به میان می آید مسیرهای دقیق مورد احتیاج قرار می گیرند . هم اکنون کاربران بسیاری از دروازه ها بهره می برند . دروازه , نامه را به سایر ماشینهای متصل ارسال می کند . اگر به دروازه ای دستیابی دارید (این موضوع را با مدیر سیستم بررسی کنید ؛ اگر توانایی ارسال و دریافت پست الکترونیکی از طریق اینترنت را دارید , این احتمال وجود دارد) , می توانید با مشخص کردن نام سیستم و نام کاربر , نامه را به دروازه بفرستید :

$ mail uunet!concubine!Kevin

که در آن uunet **نام دروازه (به طور اتفاقی** uunet **دروازه بسیار محبوبی می باشد),** concubine **نام سیستم , و** Kevin **نام کاربر می باشد . علامت تعجب (!) , ورودیها را جدا می سازد . اگر از پوسته** C **استفاده می کنید کاراکتر علامت تعجب دارای معنی خاصی می باشد (برای فهرست نمودن فرمانهای قبلی ـ تاریخچه فرمان ـ استفاده می شود) . بنابراین به منظور گریز (** escape **) از علایم تعجب باید از کاراکتر مفید وارونه استفاده کنید :**

% mail uunet\!concubine\!Kevin

فرمان بالا به پوسته C **می گوید که واقعاً قصد استفاده از علامت تعجب (!) را دارید و نمی خواهید که** csh **در سابقه خود به دنبال فرمانهای قبلی برگردد .**

طرح جدید نشانی دهی که محبوبیت آن در حال افزایش است (در اصل به خاطر افزایش محبوبیت اینترنت) نشانی دهی حوزه ای نامیده می شود . نشانی حوزه ای که دقیقاً بر خلاف مسیر دقیق ساختاردهی شده است , نام کاربر را با آدرس جفت می کند . این طرح به سبب نیاز به استانداردسازی بین المللی نشانی های پست الکترونیکی بدین شکل رشد کرده است و ساختاری سلسله ای مراتبی را برای نشاندهی ارائه می کند . در اصل , جهان حوزه های کشوری تقسیم می شود , سپس این حوزه های کشوری به حوزه های آموزشی (در آدرس با پسوند edu. نشان داده می شوند), حوزه های تجاری (در آدرس با پسوند com. نشان داده می شوند) , حوزه های دولتی (در آدرس با پسوند gov. نشان داده می شوند), و حوزه های دیگر تقسیم می گردند . صدها و صدها حوزه وجود دارند که هر روز تعداد آنها در حال افزایش است .

خواندن نشانی حوزه بسیار ساده است . در نشانی :

reichard@mr.net

reichard **به کاربر ، در حالی که** mr.net **به حوزه اشاره دارد . اسامی کاربر و حوزه با نماد@ از یکدیگر جدا می شوند . به عنوان یک کاربر , نیازی نیست که از مسیر خاصی که پیام باید طی کند و یا نام دروازه آگاهی داشته باشید . با داشتن نشانی حوزه , ارسال پیام ساده است :**

$ mail reichard@mr.net

ایده اینترنت بیشتر انتزاعی و بی نظم است . اینترنت از نظر فنی مجموعه ای از شبکه های بسیاری است که برای گفتگو با یکدیگر به طریقی اداره می شوند . تمام آن چیزی که باید

به عنوان یک کاربر بدانید , نشانی پست الکترونیکی گیرنده است ؛ جزئیات اولیه اتصال سیستم به اینترنت توسط مدیر سیستم اداره می گردند .

اگر بر روی اینترنت باشید , می توانید از راه دور نیز نامه الکترونیکی دریافت کنید . برای یافتن نشانی ماشین خود , uname -n **را در پای اعلان تایپ کنید :**

$ uname -n

yoursystem

که در آن yoursystem **نام سیستم یونیکس شما می باشد که نام میزبان نیز نامیده می شود . به منظور فهرست کردن تمام سیستم هایی که می توانید به طور مستقیم با آنها ارتباط داشته باشید, فرمان** uname **را تایپ کنید :**

uname

othersystem1

othersystem2

othersystem3

که در آن othersystem **به سایر سیستم ها اشاره می کند .**

نکته :

فرمان **uname** در تمام سیستمها امکان وجود این انتخاب را فراهم نمی کند . اگر این چنین است , می توانید برای بدست آوردن شناختی از سایر سیستم هایی که به کامپیوتر شما با آنها در شبکه قرار دارد , به پرونده **/etc/hosts** نگاه کنید .

در شبکه ای محلی یا ملی , فهرست سایر سیستمها می تواند بسیار بزرگ باشد . اگر می خواهید نام سیستم خاصی را پیدا کنید و تمایلی به جستجو در فهرست بسیار بزرگی از اسامی را ندارید , از uname در ترکیب با grep استفاده کنید :

```
$ uname |grep othersystem121
othersystem121
```

اگر نام سیستم موردنظر بازگردانده شود , می توانید به فردی که در آن سیستم دارای حساب است , پست الکترونیکی بفرستید . علاوه بر این , اگر به اینترنت متصل باشید , به شرط آگاهی از نشانی دقیق گیرنده , می توانید پیامهایی را به افراد در اینترنت ارسال کنید .

ایجاد نامه

ایجاد نامه بسیار ساده است برای ایجاد پیام کوتاهی از صفحه کلید , نامه را با نام گیرنده ترکیب کنید :

```
$ mail erc
Subject: test
This , too , is a test.
```

مانند همیشه ورودی را با ctrl+D پایان دهید . بعضی از برنامه های e-mail برای پایان دادن به پیام , به جای ctrl+D تک نقطه ای را بر روی خط تنها می پذیرند . اگر پیامی را به کاربری در ماشین راه دور می فرستادید , باز هم روال کار بدین شکل می بود :

```
$ mail kreichard@mcimail.com
```

با استفاده از گزینه t- می توانید پیام کسانی را به چندین کاربر بفرستید :

$ mail –t erc geisha spike

this, too,is a test.

پیام حاصل در سرآیند خودش چندین فیلد :To خواهد شد .

ارسال پرونده موجودی به عنوان متن پیام پست الکترونیکی تقریباً ساده است . بعد از ایجاد پرونده ای اسکی با استفاده از vi یا emacs , پرونده را ذخیره و آن را در خط فرمان به عنوان ورودی به فرمان بفرستید :

$ mail erc< note

که در آن note نام پرونده است .

برنامه mailx یونیکس بر کلی امکان می دهد که با استفاده از فرمان ~v , ویرایشگر متنی را در داخل mailx فراخوانی کنید . این فرمان را باید به تنهایی در خطی صادر نماید :

% mailx Kevin

~v

تا زمانی که سیستم خود را به شکل متفاوت دیگری پیکربندی نکرده اید , و ویرایشگر متن پیش گزیده vi خواهد بود . با استفاده از فرمانهای vi خواهد بود . با استفاده از فرمانهای vi پیام خود را ویرایش و آنگاه با فرمان zz (ذخیره و خروج) از vi خارج شوید . آنگاه به برنامه mailx برمی گردید که در آن , تک نقطه ای در خطی تنها به پایان می دهد .

کاربران system v Release 4 می توانند با به کار بردن گزینه m- پرونده هایی دودویی (مانند برنامه ها) را به عنوان پیامهای پستی ارسال کنند :

$ mail –m binaryfile ere

که در آن binaryfile نام پرونده دودویی مورد نظر برای ارسال می باشد . در سرآیند پیام حاصل , خطی نشان می دهد که پرونده دودویی است .

با پیامهای خود چه کار کنم ؟

بعد از خواندن پیام , پوسته اعلان متفاوتی به شکل زیر را نمایش می دهد و پاسخی مرتبط با برنامه mail را درخواست می کند :

?

در این نقطه اعمال بسیاری می تواند صورت بگیرد , مفیدترین انتخابها در جدول زیر فهرست می شوند:

فرمان	نتیجه
Return	پیام بعدی را چاپ می کند .
-	پیام قبلی را چاپ می کند .
d	پیام فعلی را حذف می کند .
dN	پیام شماره **N** را حذف می کند .
dp	پیام فعلی را حذف می کند و به پیام بعدی می رود .
uN	پیام **N** را بازیابی می کند .
s filename	پیام را در پرونده **filename** ذخیره می کند . اگر **filename** مشخص نشود , پیام در **$HOME**
w filename	پیام را بدون اطلاعات سرآیند در پرونده **filename** ذخیره می

	کند . اگر **filename** مشخص نشود , پیام در **$ home\mbox** ذخیره می شود .
?	فرمانهای پستی را فهرست می کند .

ذخیره کردن پیامها

همانطوری که در جدول ۱ـ۸ می بینیم , ذخیره کردن پیام به سادگی تایپ فرمان زیر می باشد :

?s

اگر اسم پرونده ای را مشخص نکنید , پیام در $HOME/mbox ذخیره می شود . اگر پیامهای بسیاری دریافت نمی کنید , مسئله ای نیست که همگی آنها را در یک پرونده ذخیره نمایید . اما اگر پیامهای بسیاری در موضوعات متنوعی دریافت می دارید , بهتر است که کارهای پستی خود را سازمان یافته کنید .

در نظر بگیرید که با کاربردی به نام erc در حال کار بر روی پروژه ای می باشید , و می خواهید که تمام پیامهای او را در یک پرونده نگه دارید . این کار را با گزینه s در اعلان ? انجام دهید :

? s erc

که در آن erc نام پرونده حاوی پیامهای پستی او می باشد . هنگامی که برای اولین بار این کار را انجام می دهید , پوسته پرونده ای به نام erc ایجاد می کند . کاربردهای بعدی , پیامهای پستی را به پرونده erc موجود اضافه خواهند کرد .

برای خواندن این پرونده , از فرمان mail به همراه گزینه f- استفاده کنید :

$ mail –f erc

سایر بسته های نرم افزاری پستی

رابط mail بیش از اندازه ابتدایی می باشد . دسته ای از برنامه های پستی جدید ظاهر شده اند که بعضی نرم افزار تجاری و بعضی رایگان هستند و هدف هر کدام ساده تر نمودن زندگی برای کاربر می باشد . این برنامه ها در قلمروی تجاری poste و Z - Mail و از میان برنامه های پستی الکترونیکی رایگان xmail, xmh, mh, mush و elm می باشند .

xmh : پیشروی mh , X Window می باشد .

pine : برنامه پست الکترونیکی معروف دیگر می باشد .

xmail : یک برنامه پستی X Window می باشد . سیستم های OpenWindows سولاریس با برنامه پست الکترونیکی گرافیکی به نام mailtool عرضه می گردند .

فایده نرم افزار رایگان در هزینه می باشد . زیان آن نیز این است خودتان (یا مدیر سیستم) مسئول نگهداری نرم افزار می باشید .

دریافته ایم که بهترین برنامه پست الکترونیکی , رایگان نیز می باشد . این برنامه که elm (electronic mail) نام دارد , رابطی با استفاده ساده را بر روی برنامه استاندارد mail فراهم می آورد .

با استفاده از کلیدهای جهت دار صفحه کلید خود می توانید پیامی را انتخاب کنید و با فشردن Return پیامی را می خوانید . elm بسیار ساده , سریع و آسان است . کمک حین کار که با تایپ علامت سوال (؟) در دسترس قرار می گیرد , راهنمای بسیار خوبی در استفاده از این برنامه می باشد .

یوزنت , اینترنت و world wide web

بحث در مورد استفاده از شبکه و ارتباطات در یونیکس با این گفتار درباره یوزنت , اینترنت و world wide web پایان می یابد .

هر کدام از این سه ابزار به جای خودش می تواند مفید باشد : یوزنت برای خواندن گروههای خبری , اینترنت برای بدست آوردن پرونده ها و اطلاعات مفید و world wide web برای متصل ماندن در این دوران از اتصال پذیری .

یوزنت و گروه های خبری

یوزنت, سیستم پیام دهی جهانی است که در آن می توان ریشه های اینترنت را یافت . یوزنت در حین این که از نظر فنی بخشی از اینترنت را در بر دارد , مناسب پست الکترونیکی و توزیع گروه خبری می باشد. هزاران کامپیوتر در سرتاسر جهان در شبکه ای

به نام یوزنت متصل می شوند . یوزنت شبکه ای عمومی از ماشینهای متصل یونیکس و غیر یونیکس می باشد که با ارسال اطلاعات به شرکتها , مدارس , دانشگاه ها , دولت , آزمایشگاههای تحقیقاتی و افراد اختصاص داده می شود. یوزنت , خدمات گوناگونی انجام می دهد , اما شاید معروف ترین سرویس آن در رابطه با گروه های خبری می باشد . گروه خبری , بحث و گفتگویی درباره موضوعات مختلف , از محاسبات تا جامعه شناسی و از بومرنگها تا دایناسورها می باشد . در حقیقت هزاران گروه خبری یوزنت وجود دارند . بعضی کم اهمیت هستند , در حالی که بعضی دیگر فقط مورد علاقه مجموعه کوچکی از کاربران می باشند :

این گروههای خبری به رده هایی تقسیم بندی می شوند تا کاربران به شکل بهتری بتوانند از میان حجم انبوه اطلاعاتی که هر روزه وارد می شود , مطالب مورد علاقه خود را برای مطالعه انتخاب کنند. یوزنت در حین این که می تواند منبع اطلاعاتی قدرتمندی باشد , اما چون تقریباً هر فردی امکان دستیابی به یوزنت دارد , عقاید و نظریات گمراه کننده , نادقیق , مغرضانه , و در کل بی معنی بسیار زیادی را در آن خواهید یافت . بنابراین آنچه که می خوانید زیادی جدی نگیرید . در کل , هر چقدر گروه فنی تر باشد , اطلاعات ارائه شده بهتر خواهد بود .شکل کلی نام گروه خبری ساده است : نام رده بندی که بعد از آن پسوند توصیفی قرار می گیرد . به عنوان مثال , نام گروه خبری اختصاص یافته به سوالاتی در رابطه با سیستم عامل یونیکس comp.unix.questions می باشد . به استفاده از نقطه ها جهت جدا

نمودن عناصر این نام توجه کنید. گروه های خبری می توانند باز یا محدود نیز باشند . در گروه های خبری باز هم فرد می تواند خبری را پست کند , در حالی که گروه های خبری محدود , قبل از رسیدن مراسلات پستی به عموم , فردی آنها را بازبینی می کند . همان طوری که حدس می زنید , گروه های خبری محدود قابل اطمینان و مفید تر می باشند .

خواندن و نوشتن اخبار

اگر چه که تمام اقلام خبری پرونده های متنی می باشند و در تئوری می توانند توسط vi **یا** emacs **خوانده شوند , اما بسیاری از آنها نیز در پرونده های جداگانه قرار دارند که خواندن آنها امکان پذیر نیست . در چنین وضعیتی نوعی نرم افزار به نام خوانندگان اخبار مطرح می شود . این خوانندگان اخبار کمک می کنند که خبرهای رسیده را بر حسب درجه های متغیر سودمندیشان مرتب کنید و از میان صد ها پرونده جدید که هر روزه ظاهر می شوند , پرونده های مورد نظرتان را بخوانید . بهتر است که پیغامهای مورد علاقه و توجه خود را بخوانید و بقیه را نادیده بگیرید . هیچ راهی برای خواندن هر پیغام وارد شونده وجود ندارد به جز اینکه تمام روز را در جلوی کامپیوترتان بگذرانید .**

هر سیستمی خواننده اخبار مخصوص به خودش را دارد . محبوب ترین برنامه های خواننده خبر عبارتند از :

Readnews**, برنامه قدیمی خطی که قدرت محدودی دارد .**

vnews , خواننده ای مانند vi که به صورت یک مورد در هر زمان , گروه های خبری را نمایش می دهد .

rn , خواننده ای با تواناییهای توسعه یافته جستجو

Xrn , نگارش گرافیکی X Window برنامه rn

Tin , خواننده ای که پیامها را بر حسب عنوان مرتب می کند .

پاسخ به اخبار نیز کار برنامه خواننده یا برنامه جداگانه ای به نام postnews می باشد .

استفاده از برنامه فوق بسیار ساده است :

$ postnews

بعد از تایپ این فرمان , مجموعه سوالاتی درباره خبری که می خواهید به آن پاسخ دهید , پرسیده می شود . نیازی نیست که به خبری خاص پاسخ دهید , اما پاسخ به گروه خبری خاص الزامی است

نکته :

تمام نگارش های یونیکس امکان وجود فرمان **postnews** را فراهم نمی آورند . بسیاری از **pnews** یا **nnpost** استفاده می کنند .

گوفرها : سازمان دهندگان اطلاعات

سرویس دهنده گوفر, روش دیگری برای سازماندهی موارد عرضه شده توسط اینترنت می باشد . گوفر که در دانشگاه مینسوتا طراحی شد , پرونده ها را توسط منوهایی سازماندهی می کند . با استفاده از گوفر می توانید پرونده هایی را از سایت FTP بازیابی کنید , به عنوان مثال , ممکن است که نام پرونده در منوی گوفر فهرست شود , اما برای شما یک سیستم پرونده سازماندهی شده و بزرگ می باشد. زیبایی گوفر در توانایی کار آن در هر دو محیط مبتنی بر متن و گرافیک می باشد . در محیط مبتنی بر متن , منوهای داخل سرویس دهنده گوفر با استفاده از کلیدهای پیمایشی یا با تایپ شماره منو انتخاب می شوند . در محیط گرافیکی (مثلاً X window) , انتخابهای منو با کلیک مضاعف ماوس صورت می گیرند .

هر نگارشی از گوفر به شکل اولیه یکسانی کار می کند . با تایپ :

```
$ gopher
```

یا

```
$ xgopher
```

آن را راه اندازی کنید .

اگر سیستم به شکل مناسبی پیکربندی شده باشد , به سرویس دهنده خانگی گوفر بر روی اینترنت متصل خواهید شد (این موضوع گیج کننده است : نرم افزاری که به طور محلی استفاده می کنید گوفر نامیده می شود و به سرویس دهنده های واقع در اینترنت نیز گوفر ها می گویند) . منویی شبیه به این را خواهید دید :

1. Minnesota Regional Network Gopher Hotel.

2. Minnesota Regional Network

3. Object Database Management Group (gopher.odmg.org,2073)

4. Minnesota Datametrics corporation (gopher.mndata.com,2074)

5. Jostens Incorporated (gopher.jostens.com,2071)

این منو , کار بزرگی برای ما انجام نمی دهد . بعد از انتخاب Minnesota Regional Network **مجموعه بعدی منو پیدا می شود که امکانات زیادی را در اختیارمان قرار می دهد.**

دستیابی به وب

بیشتر آوازه اینترنت در world wide web **و ابزارهای دستیابی به آن ـ موزاییک,** Netscape **ـ و خانواده آنها می باشد . و واقعاُ این تلاش برای متوجه ساختن انظار عمومی قابل توجیه است .** world wide web **در بهترین وضعیت ممکن راهی بسیار کارا و هیجان انگیز برای توزیع اطلاعات است .**

به منظور دستیابی به world wide web **به نظر انداز وب نیاز خواهید داشت . مشهورترین نظر اندازه وب موزاییک است که ابتدا توسط مرکز ملی کاربردهای سوپر کامپیوتر** (NCSA) **در دانشگاه ایلی نویز عرضه گردید . در دنیای یونیکس /** X window **رقیبان زیادی برای موزاییک وجود دارند , که از جمله می توان به موارد زیر اشاره نمود :**

GWHIS **(پیاده سازی تجاری موزاییک از** Quadralay **)**

Enhanced Mosaic **(نگارش تجارتی موزاییک از فروشندگان مجدد مختلف)**

Lynx **(نظرانداز متنی که در گروه نرم افزار رایگان قرار دارد)**

tkwww **(نظرانداز وب که در** Tcl/Tk **نوشته شده است و در گروه نرم افزارهای رایگان قرار دارد)**

Viola **(نظراندازی که در گروه نرم افزارهای قرار دارد)**

Netscape Navigator **(نظر اندازی که به هر دو شکل رایگان و تجارتی از سوی شرکت** Netscape Communications **در دسترس قرار می گیرد .**

نکته :

به منظور استفاده از نظر انداز وبی مانند Netscape , سیستم یونیکس باید به شکل مناسبی برای اتصال مستقیم به اینترنت پیکربندی شود . اگر کاربری تک می باشید , نیاز خواهید داشت که از طریق حساب شماره گیری یا اتصال سریع TCP/IP به اینترنت متصل شوید .

می توانید Netscape **را با وارد نمودن فرمان زیر در پنجره** xterm **راه اندازی کنید :**

```
$ netscape
```

هنگامی که Netscape **شروع به کار می کند , به طور مستقیم به صفحه خانگی** Netscape Communications **متصل می شود .**