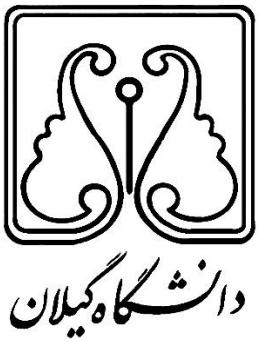


دانشگاه گیلان

دانشکده فنی

گروه مهندسی کامپیوتر

گزارش کارآموزی

پیاده سازی سیستم توزیع شده با استفاده از MPI

نام دانشجو: پارسا عباسی

شماره دانشجویی: ۹۴۰۱۲۲۶۹۰۰۶

تاریخ کارآموزی: تابستان ۱۳۹۷

استاد راهنما: دکتر حمیدرضا احمدی فر

سرپرست کارآموزی: مهندس جواد پورمصطفی

محل کارآموزی: آزمایشگاه نرم افزار – دانشکده فنی – دانشگاه گیلان

فهرست مطالب

چکیده

در این گزارش به شرح مفاهیم، ابزارها و نحوه ی پیاده سازی یک سیستم توزیع شده با استفاده از کامپیوترهای شخصی رایج پرداخته خواهد شد. این پیاده سازی بر روی مدل رابط عبور پیام (MPI) شکل خواهد گرفت و خروجی آن تقسیم وظایف محاسباتی برنامه های پیشرفته بر روی تمام سیستم های مورد آزمایش و در نتیجه اجرای آنها بصورت موازی خواهد بود.

در طول رسیدن به هدف یعنی پیاده سازی یک سیستم توزیع شده با قابلیت اجرای برنامه های موازی، از ابزارها و پروتکل های مختلفی استفاده خواهد شد که هر کدام از آنها سبب آشنایی با حوزه های متنوعی نظیر شبکه، سیستم عامل، سخت افزار و... خواهد گردید.

فصل اول
معرفی دانشگاه گیلان

۱-۱- مقدمه

در این فصل به معرفی مختصر دانشگاه گیلان، دانشکده فنی، گروه مهندسی کامپیوتر و آزمایشگاه نرم افزار به عنوان محل کارآموزی پرداخته خواهد شد.

۱-۲- تاریخچه

دانشگاه گیلان از سال ۱۳۴۶ با عنوان مدرسه عالی بازرگانی فعالیت آموزشی خود را آغاز نمود. و در سال ۱۳۵۳ به تصویب شورای گسترش آموزش عالی رسید و در سالهای ۵۵ – ۱۳۵۴ در چارچوب قرارداد بین دولتهای ایران و آلمان غربی سابق، دانشگاه گیلان ، تاسیس و از سال ۱۳۵۶ فعالیتهای آموزشی خود را با پذیرش ۱۵۵ دانشجو در ۹ رشته تحصیلی آغاز نمود.

در سال ۱۳۵۶ مدرسه عالی مدیریت و مدرسه عالی بازرگانی که تنها مراکز آموزش عالی در استان گیلان بودند با دانشگاه گیلان ادغام و فعالیت های آموزشی آن تا پایان سال تحصیلی ۱۳۵۸ با تعداد ۶۰۵ نفر دانشجو در ۱۴ رشته تحصیلی ادامه یافت.

پس از پیروزی انقلاب اسلامی، دانشگاه گیلان بطور مستقل و بدون وابستگی به آلمان، فعالیتهای آموزشی خود را در ۸ رشته تحصیلی و با حدود ۵۰۰ دانشجو از سرگرفت و با رشد و گسترش آموزش عالی در کشور، دانشکده های علوم پایه ، فنی،علوم کشاورزی،علوم انسانی و پزشکی ساخته شد و پس از آن نیز دانشکده تربیت بدنی و علوم ورزشی، دانشکده منابع طبیعی و معماری و هنر تاسیس شدند.

دانشگاه گیلان هم اکنون در ۶۴ رشته گرایش کارشناسی، ۱۶۶ رشته گرایش کارشناسی ارشد، ۹۳ رشته گرایش دکتری و در مجموع در ۳۲۳ رشته گرایش دانشجو می پذیرد.

این دانشگاه در حال حاضر با ۱۰ دانشکده و یک واحد پردیس و دو پژوهشکده (حوضه آبی دریای خزر و گیلان شناسی) با ۵۹۵ عضو هیأت علمی،حدود ۱۷۰۰۰ هزار دانشجو (حدود ۹ هزار دانشجوی دختر و ۷ هزار دانشجوی پسر) و به عنوان بزرگترین مراکز آموزش عالی در منطقه شمال کشور مشغول به فعالیت های آموزشی و پژوهشی است و سالانه حدود ۶۰۰۰ هزار نفر دانشجو را در ۴۹۴ رشته – گرایش در مقاطع تحصیلی کارشناسی، کارشناسی

ارشد و دکتری برای دوره های روزانه ، شبانه ، پردیس و مجازی پذیرش می نماید و این دانشگاه با اعضای اتحادیه دانشگاه های دولتی کشور های حاشیه دریای خزر ، برخی از دانشگاه های کشورهای جنوب شرق آسیا و اتحادیه اروپا ارتباطات علمی و تبادل استاد و دانشجو دارد.

قرار گرفتن در جمع دانشگاه های برتر سه نظام بین المللی و رتبه بندی جهانی تایمز، لیدن، شانگهای و دسته دانشگاه های یک درصد برتر دنیا و ده دانشگاه برتر جامع کشور جزئی از توانمندی های دانشگاه گیلان است. [۱]

۱-۳- اعضای هیأت رئیسه

- رئیس دانشگاه: دکتر احمد رضی
- معاون آموزشی وتحصیلات تکمیلی: دکتر فرهاد شیرینی
- معاون پژوهشی و فناوری: دکتر ضیاء الدین میرحسینی
- معاون دانشجویی: دکتر علی باستی
- معاون فرهنگی و اجتماعی: دکتر فاطمه کریمی
- معاون اداری، مالی و مدیریت منابع: دکتر حسن قلی زاده [۱]

۱-٤- معرفی دانشکده فنی

دانشکده فنی در کیلومتر ۵ جاده رشت به قزوین و در پردیس اصلی دانشگاه قرار دارد. این دانشکده از سال ۱۳۵۷ فعالیت آموزشی خود را در ۳ رشته مکانیک، عمران و برق آغاز کرد.

دانشکده فنی در حال حاضر دارای تعداد ۱۱۲ عضو هیات علمی است که

۷ نفـــر دارای مرتبه استـــــادی

۱۶ نفـــر دارای مرتبه دانشیاری

۶۹ نفـــر دارای مرتبه استادیاری

۲۰ نفـــر دارای مرتبــه مربی می باشند و خدمات علمی را به طریق تدریس و پژوهش ارایه می دهند.

تعداد کل دانشجویان این دانشکده در نیمسال اول ۹۵-۹۴دارای ۳۳۵۶ نفر می باشد. از این تعداد

۸۲۹ نفر دانشـــــجو در دوره های ارشـــــد ، دکتـــــــــری و

۲۳۴۶ نفر دانشجو در دوره کارشناسی روزانه و شبانه می باشد.

این دانشکده در حال حاضر دارای رشته گرایش های زیر است :

۳ رشته دکتری با گرایشهای :

مهندسی مکانیک: (مکانیک طراحی کاربردی)،(مکانیک – طراحی کاربردی دینامیک جامدات)،(مکانیک – طراحی کاربردی دینامیک – کنتل و ارتعاشات)،(مکانیک – تبدیل انرژی)

مهندسی عمران: (عمران – سازه)، (عمران – خاک وپی)

مهندسی نساجی

۶ رشته کارشناسی ارشد با گرایشهای:

مهندسی عمران: (سازه های دریایی، خاک پی، سازه، راه و ترابری)

مهندسی مکانیک: (طراحی کاربردی ، تبدیل انرژی)

مهندسی برق: (برق –قدرت)،(برق الکترونیک دیجیتال)،(برق الکترونیک میکروالکترونیک)،(برق مخابرات–میدان)،(برق قدرت–سیستمای قدرت)،(برق قدرت ماشین های الکتریکی و الکترونیک قدرت)،(برق– مخابرات سیستم)،(برق–الکترونیک)

مهندسی شیمی: (شیمی، شیمی فرآیندهای جداسازی)

مهندسی نساجی: (شیمی نساجی و علوم الیاف)

مهندسی کامپیوتر: (نرم افزار)

۶ رشته کارشناسی با گرایشهای:

مهندسی مکانیـــک: (حرارت سیالات، طراحی جامدات، سیتم های دینامیکی و کنترل)

مهندسی عمـــران: (عمران، عمران –عمران)

مهندسی نساجی: (علوم الیاف)

مهنـــدسی شیـمی: (مهندسی شیمی، مهندسی شیمی–صنایع غذایی)

مهندسی بـــــرق: (مهندسی برق، الکترونیک – قدرت، برق – قدرت)

مهندسی کامپیوتر: (نرم افزار – سخت افزار) [۲]

۱-۵- معرفی گروه مهندسی کامپیوتر

گروه مهندسی کامپیوتر دانشگاه گیلان از مهرماه ۱۳۸۶ با پذیرش ۴۰ دانشجو در یک رشته کارشناسی مهندسی کامپیوتر گرایش نرم افزار دوره روزانه و با ۳ هیأت علمی و یک کارشناس فعالیت خود را آغاز نمود و در حال حاضر دارای رشته های کارشناسی مهندسی کامپیوتر گرایشهای نرم افزار، سخت افزار و کارشناسی ارشد مهندسی نرم افزار و کارشناسی ارشد معماری سیستم های کامپیوتری می باشد. در کارشناسی گرایش نرم افزار دارای ۱۹۷

دانشجو، در گرایش سخت افزار ۱۰۰ دانشجو و در رشته ارشد نرم افزار ۳۲ دانشجو و در کارشناسی ارشد معماری سیستم های کامپیوتری ۱۶ دانشجو مشغول به تحصیل هستند. [۳]

۱-٦- مدیران گروه از ابتدای تأسیس تاکنون

- دکتر سیدابوالقاسم میرروشندل – از تاریخ ۱۳۹۷/۰۶/۱۱ تاکنون
- دکتر حمیدرضا احمدی فر – از تاریخ ۱۳۹۷/۰۶/۰۱ تا ۱۳۹۷/۰۶/۱۰
- دکتر اسدالله شاه بهرامی – از تاریخ ۱۳۸۹/۰۵//۰۱ تا ۱۳۹۳/۰۵/۳۱
- دکتر مهرگان مهدوی یکتایی رودسری – از تاریخ ۱۳۸۶/۰۷/۰۱ تا ۱۳۸۹/۰۴/۳۱ [۳]

۱-۷- معرفی محل کارآموزی - آزمایشگاه نرم افزار

این آزمایشگاه به ظرفیت ۱۶ دانشجو و مجهز به ۱۶ کامپیوتر و پرده نمایش و ویدئو پروژکتور بوده و جهت استفاده دروس کارگاه کامپیوتر، آزمایشگاه‌های پایگاه‌داده‌ها و سیستم‌های عامل و... در نظر گرفته شده است. [۳]

فصل دوم
کارهای انجام شده در دوره کارآموزی

۲-۱- مقدمه

در ابتدای این فصل، به بررسی برخی مفاهیم و مدل ها و ابزارهای مورد استفاده در کار خود خواهیم پرداخت. از نیاز به سیستم های با عملکرد قوی خواهیم گفت و سپس به شرح پیدایش مدل استاندارد این کار و آشنایی با یکی از پیاده سازی های آن که در این آزمایش از آن استفاده خواهد شد، می پردازیم.

۲-۱-۱- HPC[۱] چیست؟

محاسبات عملکرد بالا یا به اختصار HPC به بهره گیری از فرآیندهای موازی جهت اجرای کارآمد، قابل اطمینان و پرسرعت برنامه های پیشرفته گفته می شود. این اصطلاح خصوصاً برای سیستم هایی به کار می رود که عملکردی بیشتر از یک ترافلاپ[۲] یا 10^{12} عمل ممیز شناور در ثانیه داشته باشند. اصطلاح HPC گهگاه به عنوان مترادف واژه ی سوپر کامپیوتینگ[۳] استفاده می شود، هرچند که از نظر تکنیکی یک ابررایانه، سیستمی است که می تواند کارها را با نرخی نزدیک به بالاترین نرخ عملکردی که تاکنون کامپیوترها به آن دست یافته اند، اجرا کند. برخی ابررایانه‌ها با عملکردی بیش از یک پتافلاپ[۴] یا 10^{15} کار می کنند.

بیشترین و رایج ترین کاربران سیستم های HPC محققان علوم، مهندسان و موسسات علمی هستند. برخی آژانس های دولتی خصوصاً نظامی نیز از این سیستم ها جهت حل مسائل پیچیده بهره می برند. [۴]

HPC چندین تکنولوژی مختلف از جمله معماری کامپیوتر، الگوریتم ها، الکترونیک، سیستم های نرم افزاری و... را در زیر یک سایبان واحد گردهم آورده تا به حل کارآمد و سریع مسائل پیشرفته بپردازند. یک سیستم HPC بسیار کارآمد نیازمند یک پهنای باند وسیع و شبکه ای با زمان تاخیر[۵] اندک است تا گره ها و خوشه های مختلفی را به یکدیگر متصل نماید. [۵]

---

۱ High Performance Computing
۲ Teraflop
۳ Supercomputing
۴ Petaflop
۵ Low-latency

۲-۱-۲- تاریخچه‌ی MPI[۶]

تا پیش از ۱۹۹۰، برنامه نویسان به اندازه ی ما خوش شانس نبوده اند و نوشتن برنامه های موازی جهت استفاده در معماری های پردازشی مختلف، برای آنها دشواری های فراوانی را در پی داشت. از این رو کتابخانه های بسیاری برای ساخت اپلیکیشن های موازی توسعه یافتند، اما همچنان یک روش استاندارد توافق شده برای این کار وجود نداشت. در این زمان ها، بیشتر اپلیکیشن های موازی در دامنه ی تحقیقات و علوم تعریف می شد. با این حال، مدل رایج و مورد استفاده ی بیشتر این کتابخانه ها، همان مدل «عبور پیام»[۷] بود. مدل عبور پیام چیست؟ این مدل تنها بدین معنی است که یک اپلیکیشن به جای اجرای صرفاً یک تکلیف، به عبور پیام ها در طول پردازش ها می پردازد. مدل عبور پیام در عمل نیز خود را به خوبی برای برنامه های موازی ثابت کرده بود. به عنوان مثال، یک فرآیند ارباب یا مستر[۸]، با ارسال یک پیام به فرآیندهای برده[۹] که حاوی توصیفی از یک کار است، آن را به برده ها تخصیص می داد. مثال دیگر، برنامه مرتب سازی ادغامی[۱۰] بود که اطلاعات را به صورت محلی مرتب کرده و سپس آنها را به فرآیندهای همسایه ارسال می کرد تا لیست های مرتب شده با یکدیگر ادغام شوند. از این رو تقریباً تمام اپلیکیشن های موازی با مدل عبور پیام قابل تعریف هستند.

از آنجایی که بسیاری از کتابخانه های این زمان از همان مدل عبور پیام و تنها با تفاوت در برخی ویژگی ها استفاده می کردند، نویسنده های این کتابخانه ها و برخی متخصصین دیگر در کنفرانس Supercomputing 1992 گردهم آمدند تا یک رابط استاندارد را برای اجرای عبوردهنده ی پیام که ما امروزه آن را «رابط عبور پیام» یا به اختصار «MPI» می نامیم تعریف کنند. این رابط استاندارد به برنامه نویسان اجازه می داد تا اپلیکیشن های موازی را به صورت پرتابل طراحی کنند تا بر روی تمام انواع معماری های موازی قابل تعریف باشد. علاوه بر این، اجازه ی استفاده از ویژگی ها و مدل های مختلفی که تا اکنون از آنها بهره می بردند، در داخل کتابخانه های کنونی نیز داده می شد.

در سال ۱۹۹۴، یک رابط کامل و استاندارد با نام «MPI-1» تعریف گردید. باید توجه داشت که MPI تنها یک تعریف برای یک رابط است. بعد از این است که توسعه دهندگان به ساخت پیاده سازی های رابط برای معماری های مخصوص به خود پرداختند. خوشبختانه تنها کمتر از یک سال، پیاده سازی های کاملی برای مدل MPI توسعه یافته و در دسترس قرار گرفت. بعد از اولین پیاده سازی ها، این مدل بسیار گسترده شد و تا امروز نیز از

---

[۶] Message Passing Interface
[۷] Message Passing
[۸] Master
[۹] Slave
[۱۰] Merge Sort

آن در پردازش های موازی استفاده می شود. از جمله پیاده سازی های محبوب MPI که امروزه مورد استفاده قرار می گیرند، می توان به OpenMPI و MPICH2 اشاره کرد. [۶]

۲-۱-۳- معرفی MPICH و MPICH2

MPICH یک پیاده سازی قابل حمل وسیع و با عملکرد بالا از استاندارد (MPI-1، MPI-2 و MPI-3) رابط عبور پیام (MPI) است. اهداف MPICH را می توان در موارد زیر خلاصه نمود :

۱. فراهم سازی یک پیاده سازی مدل MPI که بصورت کارآمد از محاسبات و پلتفرم های ارتباطی مختلف نظیر خوشه های صنعتی[۱۱] (سیستم های دسکتاپ، سیستم های حافظه مشترک، معماری های چند هسته ای)، شبکه های پرسرعت (اترنت ۱۰ گیگابیت، InfiniBand، Myrinet، Quadrics) و سیستم های محاسباتی اختصاصی (Cray ،Blue Gene) پشتیبانی کند.
۲. فراهم سازی امکان تحقیقات پیشرفته تر و جدیدتر در حوزه MPI با استفاده از یک چارچوب مدولار آسان-برای-گسترش[۱۲].

همچنین MPICH بصورت اوپن سورس توسعه یافته و بصورت آزاد و رایگان قابل دسترس است. این پکیج بر روی چندین پتلفرم مختلف نظیر لینوکس (IA32 و x86-64)، مک او اس ده (PowerPC و Intel)، سولاریس و ویندوز مورد آزمایش قرار گرفته است.

توسعه ی MPICH از همان سال استانداردسازی MPI یعنی سال ۱۹۹۲ آغاز شد. این پیاده سازی اورجینال بر پایه سیستم قابل حمل Chameleon شکل گرفت تا یک لایه پیاده سازی سبک-وزن[۱۳] فراهم گردد (اکنون مشخص است که نام MPICH از دو واژه ی MPI و CHameleon شکل گرفته است).

حوالی سال ۲۰۰۱، توسعه ی یک پیاده سازی جدید به نام MPICH2 آغاز گردید. تفاوت این نسخه با نسخه ی اول، در پیاده سازی ویژگی های اضافه ای است که در استاندارد MPI-2 تعبیه شده بود. آخرین نسخه ی MPICH با شماره ی 1.2.7p1 منتشر گردید. با این حال، شماره نسخه های MPICH2 مجدداً از شماره ی 0.9 راه اندازی شد و تا شماره ی 1.5 ادامه یافت. با شروع انتشار بزرگ در نوامبر سال ۲۰۱۲، این پروژه

---

[۱۱] Commodity Clusters
[۱۲] Easy-to-Extend Modular Framework
[۱۳] Light-Weight

دوباره به MPICH تغییر نام یافت و با شماره نسخه ی 3.0 آغاز به کار کرد. لازم به ذکر است آخرین نسخه ای که اکنون بصورت پایدار در دسترس است و ما نیز در کارآموزی خود از آن استفاده خواهیم کرد MPICH 3.2.1 نام دارد. [۷]

اکنون که با مفاهیم اولیه و ابزارهای مورد استفاده ی خود آشنا هستیم می توانیم به ترتیب به شرح گام های طی شده در این کارآموزی بپردازیم. لازم به ذکر است که هدف ما در این دوره، صرفاً افزایش قدرت محاسباتی با درگیرسازی پردازنده های مرکزی تمام سیستم ها است.

۲-۲- آماده سازی

به عنوان نخستین گام جهت ایجاد یک خوشه و پیاده سازی امکان پردازش موازی با استفاده از MPICH، نیاز است تا سیستم های مورد استفاده خود را انتخاب و آماده نماییم. در انتخاب خود به مواردی همچون تعداد پردازنده ها، وجود کارت گرافیک (جهت استفاده در پردازش های قوی تر در صورت نیاز)، سیستم عامل و... توجه خواهیم کرد. لازم به ذکر است به عنوان یک نمونه ی پژوهشی صرفاً ۵ سیستم، یک عدد به عنوان سرور یا مستر و ۴ سیستم دیگر به عنوان گره یا برده انتخاب خواهیم کرد.

برای پیاده سازی مدل MPI به کمک پکیج MPICH و پیش نیازهای آن ما سیستم عامل لینوکس و نسخه ی اوبونتو را برای این کار برگزیده ایم زیرا که در این زمینه از قدرت، شفافیت و امکانات بیشتری برخوردار است. از این رو به بررسی سیستم عامل کنونی سیستم های انتخابی خود خواهیم پرداخت. برخی از این سیستم ها از پیش به این سیستم عامل مجهز بوده اند اما تصمیم بر پاکسازی کامل آنها و نصب مجدد این سیستم عامل گرفته شد که از دلایل آن می توان به عدم دسترسی به رمز عبور کاربر اصلی، امکان تداخل در پیکربندی پکیج مورد استفاده ی ما با دیگر پکیج ها یا تداخل در پیکربندی شبکه و... اشاره کرد.

بنابراین در این مرحله ترجیحاً یک نسخه مشابه (16.04 LTS) از سیستم عامل Ubuntu را بر روی تمام کامپیوترهای انتخابی خود نصب خواهیم کرد. بعد از این کار، مشخصات سیستم هایی که در اختیار داریم به صورت جدول (۲-۱) خواهد بود.

جدول (۲-۱)

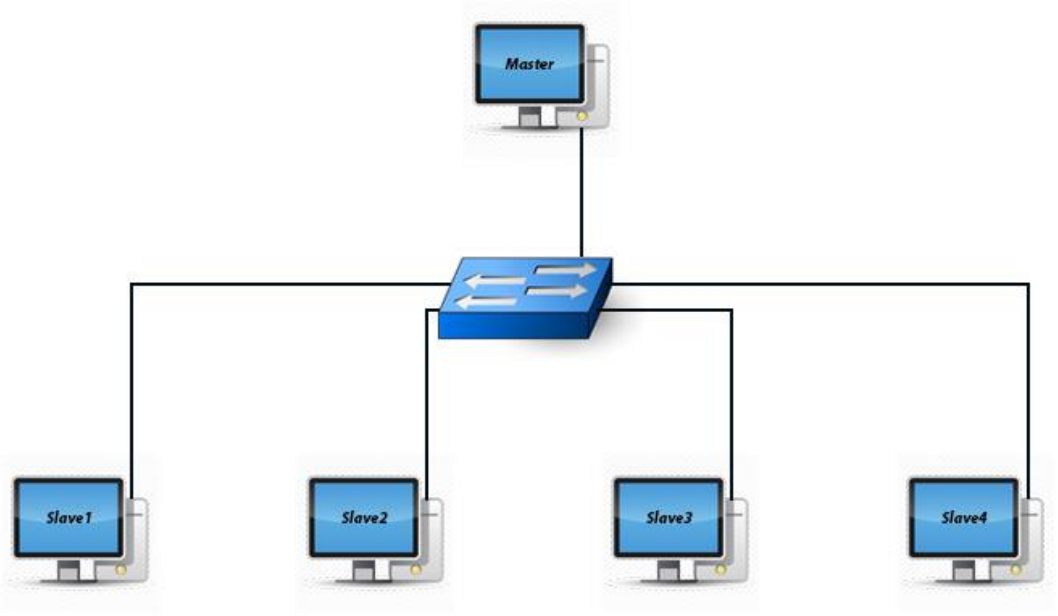
نام	شماره سیستم	فرکانس پردازنده	سیستم عامل
Master	PC-42	2.80 GHz	Ubuntu 16.04 LTS
Slave1	PC-48	2.60 GHz	Ubuntu 16.04 LTS
Slave2	PC-49	2.60 GHz	Ubuntu 16.04 LTS
Slave3	PC-60	3.10 GHz	Ubuntu 16.04 LTS
Slave4	PC-63	2.60 GHz	Ubuntu 16.04 LTS

لازم به ذکر است در هنگام نصب سیستم عامل، برای سادگی بیشتر کار در آینده، نام کاربری و رمز عبور یکسان برای تمام سیستم ها انتخاب شده است. همچنین نیاز است تا یک کاربر جدید بر روی تمام سیستم ها تعریف نماییم و مشخصات آن نیز باید بر روی همه ی آنها مشابه باشد. (نحوه ی انجام این کار در پیوست ۱ شرح داده شده است.) نام کاربری که ما برای این کاربر انتخاب کرده ایم ***mpiuser*** می باشد. علت این کار، جداسازی محیط کاری خود از کاربر اصلی یا به اصطلاح ریشه می باشد. همچنین در صورتیکه نام کاربری یکسان برای همه ی سیستم ها انتخاب نشده باشد در مراحلی نظیر راه اندازی SSH یا اجرای MPICH با مشکل مواجه خواهیم شد. بنابراین از این پس تمام کارهای خود را بر روی محیط همین کاربر یعنی ***mpiuser*** انجام خواهیم داد. اکنون تمام کامپیوترهای ما برای شروع کار آماده هستند.

۲-۳- شبکه سازی

اکنون که سیستم های ما همگی آماده ی کار هستند می توانیم به اجرای گام بعدی، یعنی برقراری ارتباط بین کامپیوترها بپردازیم. پرواضح است که برای ارسال و دریافت پیام بین کامپیوترها در متد MPI به این راه ارتباطی نیاز خواهیم داشت. برای ایجاد این ارتباط کافیست دست به ساخت یک شبکه ی محلی روی این سیستم ها بزنیم. بدین منظور از آنجایی که تمام سیستم های ما از کارت شبکه برخوردار هستند می توانیم به کمک کابل های LAN و یک عدد سوییچ[۱۴] حداقل ۵ پورته (در این مثال) این شبکه را برقرار نماییم. ساختار ارتباط فیزیکی کامپیوترها در شکل (۲-۱) قابل مشاهده است.

[۱۴] Switch



(شکل ۲-۱)

همانطور که در تصویر بالا مشخص است، هرکدام از کامپیوترها قادر خواهد بود با دیگری در ارتباط باشد. همچنین اگر صرفاً یک کدام آنها از طریق یک رابط[۱۵] شبکه ی دیگر به شبکه ی اینترنت متصل باشد یا یک کابل LAN دیگر را از مودم یا یک سیستم متصل به اینترنت دیگر به سوییچ موجود در ساختار خود متصل کنیم، آن سیستم نقش پل[۱۶] را بازی کرده و دیگر کامپیوترها نیز قادر خواهند بود از این طریق به شبکه ی اینترنت متصل گردند.

لازم به ذکر است که برای ساخت کامل این شبکه علاوه بر ارتباط فیزیکی نیاز است به پیکربندی نرم افزاری آنها نیز بپردازیم. بدین منظور ما از آی پی های ایستا[۱۷] استفاده خواهیم کرد که شرح کامل تر آن و نحوه تخصیص آی پی به یک کامپیوتر در سیستم عامل اوبونتو، درون فصل سوم گنجانده شده است.

پس از پیکربندی کامل شبکه ی خود، آی پی های انتخابی و اختصاص داده شده به هر کامپیوتر مطابق با جدول (۲-۲) خواهد بود.

---

[۱۵] Interface
[۱۶] Bridge
[۱۷] Static

جدول (۲-۲)

نام	شماره سیستم	آدرس آی پی
Master	PC-42	192.168.1.10
Slave1	PC-48	192.168.1.11
Slave2	PC-49	192.168.1.12
Slave3	PC-60	192.168.1.13
Slave4	PC-63	192.168.1.14

تا اینجا، پیکربندی شبکه بین کامپیوترها کامل شده است و هرکدام دارای آدرس آی پی منحصر به فرد هستند. با این حال از آنجایی که به خاطر سپاری آی پی هر کدام از سیستم ها و همچنین تایپ آنها احتمالاً امری وقت گیر خواهد بود، بهتر است بتوانیم تنها با استفاده از نام دلخواه خودمان به آنها دسترسی پیدا کنیم. بدین منظور کافیست نام میزبان[۱۸] هر آدرس آی پی را مشخصاً برای هر کامپیوتر تعریف نماییم. جهت آشنایی با روش انجام این کار در سیستم عامل اوبونتو به پیوست ۲ مراجعه شود. یعنی به عنوان مثال از این پس قادر خواهیم بود به جای نوشتن آدرس کامل 192.168.1.10، صرفاً با وارد کردن عبارت *master* به این گره در شبکه ی خود دسترسی پیدا کنیم.

۲-٤- تنظیم SSH[۱۹]

SSH یک پروتکل رمزنگاری شبکه است که امکان اجرای امن سرویس های شبکه را بر روی یک شبکه ی ناامن فراهم می سازد. پورت استاندارد TCP برای این پروتکل ۲۲ می باشد. یک نمونه ی شناخته شده ی این موضوع در ورود از راه دور کاربران به سیستم های کامپیوتری می باشد.

SSH با استفاده از معماری کلاینت-سرور، یک کانال امن را بر روی یک شبکه غیرامن به ارمغان می آورد. جدا از مثال معروف بالا می توان گفت هر سرویس شبکه ای قادر خواهد بود با استفاده از این پروتکل امن گردد. بیشترین و رایج ترین استفاده از این پروتکل در سیستم عامل های مشابه یونیکس می باشد اما مایکروسافت نیز از سال ۲۰۱۵ تصمیم گرفت این پروتکل را در نسخه های بعدی سیستم عامل خود یعنی ویندوز پشتیبانی کند. [۸]

[۱۸] Host Name
[۱۹] Secure Shell

و اما استفاده ی ما از SSH در این دوره به دلیل نیاز MPICH به آن و عدم درخواست رمزعبور در هر بار درخواست دسترسی کامپیوتر مستر به کامپیوترهای برده می باشد. این کار به راحتی و با ساخت یک کلید اس اس اچ[۲۰] در سرور و کپی آن به کامپیوترهای گره انجام می گیرد. هرچند که ظاهر راه اندازی این پروتکل، امری ساده به نظر می رسد اما باید با دقت انجام گیرد و کوچکترین خطایی می تواند باعث بروز مشکل در گام های بعدی شود. شرح کامل راه اندازی SSH در فصل چهارم این گزارش آورده شده است. همانطور که گفته شد، پس از پیکربندی کامل این پروتکل قادر خواهیم بود از کامپیوتر مستر بدون نیاز به وارد کردن رمزعبور کامپیوترهای برده، به آنها وارد شویم.

۲-۵- تنظیم NFS[۲۱]

NFS یک پروتکل توزیع شده سیستم فایل می باشد که توسط Sun Microsystems و در سال ۱۹۸۴ توسعه یافته است. این پروتکل به کاربرِ یک کامپیوتر کلاینت این اجازه را می دهد تا به فایل های موجود روی یک شبکه ی کامپیوتری دسترسی پیدا کند. این بدین معنی است که وقتی فایلی بر روی پوشه ی NFS در شبکه قرار گرفته می شود، تمام سیستم هایی که از طریق پروتکل NFS به شبکه متصل می شوند قادر به دسترسی به آن فایل خواهند بود. از این رو، استفاده از این پروتکل از جهت های مختلفی برایمان مفید واقع خواهد شد. اول از همه، در برخی موارد که نیاز به دسترسی به یک فایل بر روی همه ی سیستم ها داریم نیاز به دریافت یا انتقال چندباره ی آن وجود نخواهد داشت و کافیست آن را بر روی پوشه ی مشترک NFS قرار دهیم تا در دسترس تمام شان قرار گیرد. دلیل دیگر، ساختار پیاده سازی MPICH می باشد، که نیاز است تمام سیستم ها دقیقاً از یک منبع مشترک این کتابخانه استفاده کنند و فایل برنامه ای که قرار است بصورت موازی انجام شود نیز از یک مسیر مشترک بین سیستم ها فراخوانی گردد. بنابراین طبق فصل پنجم به پیاده سازی این پروتکل پرداخته و حاصل کار یک پوشه به نام nfsshare/ خواهد بود که تمام کامپیوترها قادر خواهند بود به هر فایل موجود در آن دسترسی پیدا کنند. [۹]

---

[۲۰] SSH-Keygen
[۲۱] Network File System

۲-٦- پیاده سازی MPICH

مقدمات و مفاهیم این نسخه از پیاده سازی مدل MPI در ابتدای فصل شرح داده شد و تا اینجای کار نیز پیش نیازهای آن را به صورت کامل راه اندازی کرده ایم. بنابراین اکنون می توان طبق آنچه که در فصل ششم شرح داده شده به دریافت، نصب و پیکربندی این پکیج بپردازیم. لازم به ذکر است از آنجایی که تمام این کارها را بر روی همان پوشه ی nfsshare انجام داده خواهد شد، تمام کامپیوترها قادر به دسترسی به این پکیج بوده و نیازی به نصب مجدد آن بر روی تک تک سیستم ها نخواهد بود. بنابراین پس از تکمیل راه اندازی این پکیج می توانیم به سراغ اجرای یک برنامه توسط آن و تحلیل خروجی برویم.

۲-۷- تست و بررسی

حال که همه چیز به درستی پیاده سازی شده است می توانیم به سراغ اجرای برنامه های موازی خود که با بهره گیری از مدل MPI توسعه یافته اند بپردازیم. جهت اجرای این برنامه ها که عموماً به زبان C یا C++ نوشته می شوند کافیست طبق فصل هفتم به کامپایل کردن آنها و سپس اجرا توسط MPICH بپردازیم. همانطور که در فصل هفتم شرح داده شده می توان به کمک افزونه ی htop به بررسی عملکرد سیستم و پردازنده های آن پرداخت.

به عنوان تست سیستم توزیع شده ی خود ما به اجرای دو برنامه به زبان C خواهیم پرداخت که اولی تخمین عدد پی[۲۲] و دیگری ضرب ماتریس ها [۱۰] است که از بار محاسباتی بسیار بیشتری برخوردار است.

در اجرای برنامه تخمین عدد پی آنچه که به خوبی قابل مشاهده است، تقسیم برنامه و تخصیص مناسب هر زیربرنامه به سیستم های برده می باشد. پس از آن با تغییر در تعداد پردازنده های درگیر در اجرای برنامه می توان به خوبی دریافت که با بر هم زدن این توازن و افزایش این تعداد، سرعت اجرا بیشتر شده اما دقت تخمین کاهش می یابد. همانطور که مشخص است علت این تفاوت، در بار محاسباتی بیشتر بر روی پردازنده های یک سیستم می باشد. و اما در برنامه ی دوم یعنی ضرب ماتریس ها، می توان با انتخاب ماتریس های بزرگ به وضوح عملکرد تقریباً صد در صدی تمام پردازنده ها را مشاهده کرد. نتیجه ی دوم در مثال قبل در مورد این برنامه نیز صادق است، هر چه توازن توزیع محاسبات بر روی سیستم های مختلف بیشتر باشد، خروجی نیز از دقت بالاتری برخوردار خواهد بود.

[۲۲] موجود در پوشه ی Examples پکیج MPICH

فصل سوم
پیکربندی شبکه

۳-۱- مقدمه

همانطور که در فصل دوم بدان اشاره شد، برای پیکربندی یک شبکه ی محلی بر روی کامپیوترهای موجود، از آی پی های ایستا استفاده خواهیم کرد. در سیستم عامل اوبونتو، بصورت پیش فرض هنگامیکه کارت شبکه فعال شده و به یک شبکه متصل می شود، پیکربندی آی پی آن به فرم dhcp خواهد شد و این بدین معنی است که تنظیم آی پی برای آن سیستم بصورت خودکار انجام می شود و هربار ممکن است مقدار متفاوتی را به خود اختصاص دهد. اما از آنجایی که قرار است ما کاملاً از ساختار شبکه ی خود مطلع بوده و مشخصاً بدانیم که هر کامپیوتر با چه کامپیوترهایی در ارتباط است، با تخصیص یک آی پی ایستا به آن کامپیوتر هویت یکتا خواهیم بخشید.

۳-۲- تخصیص آی پی

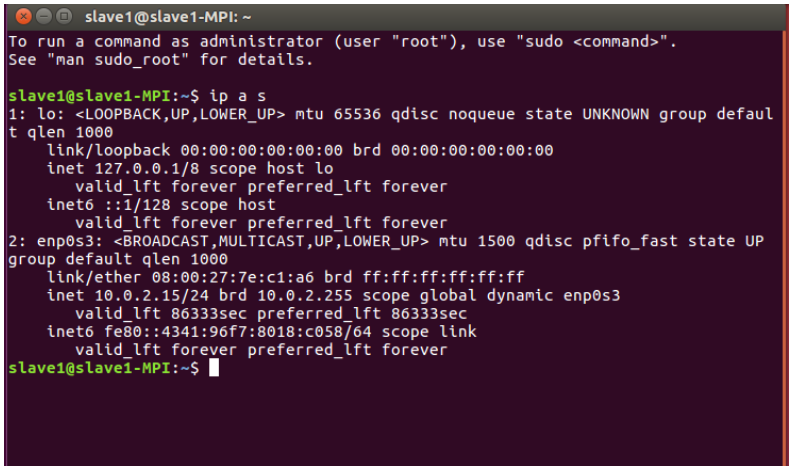
از آنجایی که کامپیوتر در اختیار ما ممکن است از چندین رابط شبکه برخوردار باشد، ابتدا باید از تعداد و نام این رابط ها اطلاع کسب نماییم. اگر این تعداد بیشتر از یک رابط بود، می توانیم از یک رابط به عنوان دسترسی به اینترنت و از دیگری به عنوان رابط شبکه ی محلی خود استفاده کنیم. بدین منظور کافیست در ترمینال اوبونتو دستور زیر را وارد کنیم تا لیست تمام رابط ها، آی پی آنها و سایر مشخصات نمایش داده شود :

```
$ ip address show
```

یا به اختصار :

```
$ ip a s
```

اکنون لیست تمام رابط ها ظاهر شده و نامی که در جلوی هر ردیف قرار داد، نمایانگر نام همان رابط می باشد. به عنوان مثال در تصویر (۳-۱) *lo* و *enp0s3* نام رابط های این سیستم می باشد و این بدین معنی است تنها یک رابط شبکه در اختیار خواهیم داشت و آن رابط *enp0s3* خواهد بود.



شکل (۳-۱)

اکنون می توانیم به اختصاص یک آدرس آی پی یکتا به این رابط بپردازیم. پیش از هر چیز باید توجه داشت که در شبکه ی محلی خود از آی پی های محدوده ی کلاس C استفاده خواهیم کرد که جزء آی پی های خصوصی می باشد و در خارج از همین شبکه معتبر نمی باشند.

جهت تغییر در پیکربندی رابط های شبکه در سیستم عامل اوبونتو کافیست به فایل *interfaces* در مسیر */etc/network* را مراجعه کنیم. با استفاده از دستور زیر می توان به اجرای این فایل در حالت ویرایش اقدام کرد :

```
$ sudo pico /etc/network/interfaces
```

اکنون می توان اطلاعات موجود در این فایل را برای اختصاص آی پی استاتیک تغییر داد. نمونه ی این تغییر برای کامپیوتر Master به صورت زیر خواهد بود :

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback


auto enp0s3
iface enp0s3 inet static
address 192.168.1.10
netmask 255.255.255.0
```

در قسمت آدرس، آی پی ایستای انتخابی خود را انتخاب خواهیم کرد. یعنی در این مثال به کامپیوتر Master خود آی پی 192.168.1.10 را اختصاص داده ایم. در همین فایل می توان به تعریف اطلاعات دیگری نظیر آدرس شبکه، آدرس پخش و... نیز اقدام کرد که در اینجا برای کار خود به آنها نیازی نخواهیم داشت.

جهت ذخیره ی تغییرات و اعمال آن بر روی کارت شبکه نیاز است تا یکبار به راه اندازی مجدد آن بپردازیم. بدین منظور می توان از دستور زیر در محیط ترمینال کمک گرفت :

```
$ sudo service networking restart
```

نکته: برای ذخیره فایل و خروج از آن در محیط pico کافیست ابتدا کلید ترکیبی Ctrl+O، سپس Enter جهت تایید ذخیره و سپس کلید ترکیبی Ctrl+X جهت خروج را فشار دهیم.

همین کار را بر روی تمام سیستم ها تکرار کرده با این تفاوت که به هر سیستم یک آی پی یکتا اختصاص می دهیم. پر واضح است که آدرس ماسک شبکه برای تمام سیستم ها یکسان تعریف می شود. [۱۱]

لازم به ذکر است از آنجایی که سیستم های موجود تنها از یک کارت شبکه برخوردار بوده اند، با حذف قابلیت *dhcp* و تعریف آی پی ایستا، امکان اتصال کامپیوتر به سایر شبکه ها نظیر شبکه اینترنت وجود نخواهد داشت. بنابراین در صورت نیاز به دسترسی به اینترنت باید هر مرتبه به تغییر اطلاعات این فایل از آی پی ایستا به *dhcp*، و برعکس آن برای برقراری شبکه محلی پرداخت.

۳-۳- روش گرفتن پینگ[۲۳]

جهت اینکه از وجود ارتباط بین تمام کامپیوترها مطمئن شویم کافیست از هر کامپیوتر به آدرس آی پی دیگر کامپیوترها پینگ گرفته شود. بازگشت و اعلام زمان پینگ به معنی برقراری صحیح ارتباط بین دو سیستم خواهد بود. برای پینگ گرفتن در سیستم عامل اوبونتو می توان به راحتی از دستور زیر استفاده کرده و تنها به جای عبارت *address*، آدرس آی پی مقصد را وارد نمود :

```
$ ping address
```

---

[۲۳] Ping

فصل چهارم
راه اندازی SSH

٤-١- مقدمه

همانطور که در فصل دوم گفته شد، جهت امکان ارتباط آزادانه سرور با گره ها (برده ها) نیاز است تا پروتکل SSH را بر روی آنها راه اندازی کنیم. بدین منظور نیاز است تا گام های شرح داده شده در این فصل با دقت طی شوند تا در ادامه ی دوره و خصوصاً هنگام راه اندازی MPICH با مشکل مواجه نشویم.

٤-٢- نصب پکیج مختص به سرور

بعد از به اتمام رسانیدن گام قبل، نوبت به نصب پکیج SSH مختص به سرور می رسد. بنابراین در کامپیوتر مستر خود وارد ترمینال شده و دستور زیر را جهت دریافت و نصب این پکیج وارد کنیم. لازم به ذکر است که بدین منظور نیاز است تا به شبکه اینترنت دسترسی داشته باشیم. اگر اینطور نیست باید طبق فصل سوم، پیکربندی رابط شبکه در فایل /etc/network/interfaces را به *dhcp* تغییر داد و پس از تمام کار مجدداً آن را به حالت قبلی یعنی *static* بازگرداند.

```
$ sudo apt-get install openssh-server
```

بهتر است بعد از دریافت پکیج بالا، یکبار با استفاده از دستور زیر به آپدیت apt-get نیز بپردازیم.

```
$ sudo apt-get update
```

٤-٣- برقراری اتصال SSH

در سیستم عامل اوبونتو، بصورت پیش فرض نسخه ی کلاینت این پروتکل بر روی سیستم نصب می شود. بنابراین هم اکنون باید بتوان بدون مشکل از یک سیستم به یک سیستم دیگر از طریق این پروتکل متصل گردید. این کار با استفاده از اجرای دستور زیر در ترمینال صورت می گیرد. به جای عبارت *ip-address* می توان آدرس کامل آی پی کامپیوتر مقصد یا نام میزبان آن را که در گام شبکه سازی تعریف شده بود وارد نمود.

```
$ ssh mpiuser@ip-address
```

بعد از وارد کردن دستور بالا، در صورتی که کامپیوتر مقصد از قبل در لیست میزبان های شناخته شده ی کامپیوتر مبداء وجود نداشته باشد، سوالی پرسیده می شود که کافیست عبارت yes را تایپ کرده و اجرا کنیم. بعد از کافیست رمزعبور سیستم مقصد را وارد کنیم (که اینجا تمام رمزها یکسان در نظر گرفته شده است). اکنون مشاهده می شود که محیط کاری ترمینال ما به محیط کامپیوتر دیگر تغییر پیدا کرده و قادر به مشاهده ی تمام فایل های سیستم مقصد و اجرای دستورات مختلف در آن هستیم. جهت خروج از این سیستم و بازگشت به محیط سیستم اصلی خودمان کافیست دستور exit را وارد نماییم.

اکنون که با نحوه ی اتصال سیستم ها از طریق SSH آشنا شدیم، نیاز است تا یکبار از هر کدام از کامپیوترهای خود به هر کدام از کامپیوترهای دیگر از این طریق متصل شویم. این کار به این دلیل است که اولاً فولدر پیش فرض ssh. بر روی همه ی کامپیوترها ایجاد گردد و ثانیاً هر کامپیوتر به لیست میزبان های شناخته شده (فایل *known_hosts* در پوشه ی *ssh*.) در هر سیستم اضافه گردد.

٤-٤- ساخت کلید SSH

اکنون که همه چیز جهت راه اندازی این پروتکل آماده است به سراغ کامپیوتر مستر یا به اصطلاح سرور رفته و با استفاده از دستور زیر یک کلید رمزنگاری شده SSH ایجاد می کنیم. این کلید همان راه حل ارتباط امن بین سیستم ها است که در گام های بعد از آن بهره خواهیم برد. لازم به ذکر است که این کلید باید در داخل پوشه ی ssh. کامپیوتر مستر ساخته شود پس اول از هر چیز نیاز است تا وارد این مسیر شویم.

```
$ cd .ssh
```

نکته: باید توجه داشت که در هنگام ساخت کلید، از دسترسی sudo استفاده نشود.

```
/.ssh$ ssh-keygen -t rsa
```

اکنون اگر به مشاهده ی فایل های درون این پوشه بپردازیم متوجه دو فایل جدید به نام های id_rsa و id_rsa.pub خواهیم شد که فایل دوم همان کلید عمومی قابل انتقال به کامپیوترهای برده است.

٤-٥- کپی کلید عمومی

اکنون نیاز است تا کلید عمومی ساخته شده در گام قبل را به درون کامپیوترهای برده (یا همان گره در شبکه) انتشار دهیم. بدین منظور کافیست در داخل کامپیوتر مستر، دستور زیر را وارد نماییم، تا یک کپی از فایل کلید عمومی به درون یک کامپیوتر دیگر (در اینجا برای مثال slave1) انتقال داده شود :

```
/.ssh$ scp id_rsa.pub mpiuser@slave1:~/.ssh/id_rsa.pub
```

همین کار را برای تمام سیستم ها انجام خواهیم داد. یعنی کافیست در دستور بالا به جای عبارت slave1، نام میزبان های دیگر (slave2, slave3, ...) را نوشته و اجرا کنیم. با این کار یک کپی از کلید عمومی بر روی تمام سیستم ها وجود خواهد داشت.

٤-٦- ساخت فایل کلیدهای مجاز

حال نوبت به آخرین گام یعنی ساخت فایل کلیدهای مجاز[۲۴] در درون کامپیوترهای برده می رسد. با این کار، هنگامیکه کامپیوتر مستر درخواست دسترسی به دیگر کامپیوترها از طریق SSH را داشته باشد، بدون نیاز به رمزعبور این امر میسر می شود زیرا که کلید این سیستم از پیش در درون آنها تعریف شده است. بدین منظور کافیست بر روی هرکدام از کامپیوترهای برده به پوشه ی ssh. رفته و دستور زیر را اجرا نماییم :

```
/.ssh$ cat id_rsa.pub >> authorized_keys
```

حال اگر به مشاهده ی فایل های درون این پوشه بپردازیم، متوجه یک فایل جدید به نام ***authorized_keys*** خواهیم شد که اطلاعات موجود در آن دقیقاً مشابه اطلاعات فایل ***id_rsa.pub*** می باشد. [۱۲]

---

[۲۴] Authorized-Keys

٤-۷- تست عملکرد SSH

اکنون طبق هدفی که در پیش داشتیم، باید بتوان از کامپیوتر مستر بدون نیاز به رمز عبور به سایر سیستم ها متصل گردید. جهت اطمینان کافیست درون کامپیوتر مستر، دستور زیر را هربار برای هرکدام از دیگر کامپیوترها اجرا نماییم : (لازم به ذکر است همانطور که پیش تر نیز گفته شد، جهت خروج از اتصال، از دستور exit استفاده می شود)

```
$ ssh mpiuser@slave1
```

فصل پنجم
راه اندازی NFS

۵-۱- مقدمه

طبق آنچه که در فصل دوم شرح داده شد، در اینجا به استفاده از پروتکل رایج NFS جهت ساخت یک پوشه بر روی شبکه خواهیم پرداخت و با این کار تمام کامپیوترها قادر به دسترسی به این پوشه و فایل های داخل آن خواهند بود.

۵-۲- نصب پکیج NFS روی سرور

اول از هر چیز نیاز است تا پکیج مربوط به سرور پروتکل NFS را بر روی کامپیوتر مستر خود دریافت و نصب نماییم. بدین منظور از دستور زیر استفاده خواهیم کرد :

```
$ sudo apt-get install nfs-kernel-server portmap
```

۵-۳- ساخت پوشه ی مشترک در سرور

اکنون یک پوشه بر روی کامپیوتر مستر خواهیم ساخت و این همان پوشه ای خواهد بود که قرار است تمام سیستم ها بتوانند از طریق پروتکل NFS آزادانه بدان دسترسی پیدا کنند. نامی که ما برای این پوشه برگزیده ایم /nfsshare می باشد.

```
$ sudo mkdir /nfsshare
```

همچنین از آنجایی که قرار نیست صاحب این پوشه به سیستمی غیر از سیستم مستر تغییر کند، نیاز است تا حالت دسترسی آن را با استفاده از دستور زیر تغییر دهیم :

```
$ sudo chown nobody:nogroup /nfsshare
```

۵-٤- تعریف پوشه به عنوان خروجی

حال نیاز است که این پوشه را در لیست خروجی پروتکل تعریف کنیم. بدین منظور با استفاده از کد زیر فایل exports را در محیط ویرایشگر pico باز خواهیم کرد.

```
$ sudo pico /etc/exports
```

اکنون خط زیر را به انتهای فایل اضافه کرده و تغییرات را ذخیره خواهیم کرد.

```
/nfsshare *(rw,sync)
```

باید توجه داشت که علامت (*) در دستور بالا به معنی هر آی پی است. در حالت عادی و البته امن تر می توان آی پی سیستم هایی که قصد داریم بتوانند به این پوشه دسترسی پیدا کنند را مشخصاً و جداگانه تعریف کنیم. اما استفاده از این علامت، بدین معنی است که هر سیستمی قادر به دسترسی به این پوشه خواهد بود.

۵-۵- نصب پکیج NFS روی کلاینت ها

حال باید نسخه ی مختص به کلاینت پکیج NFS را بر روی تمام کامپیوترهای برده دریافت کرده و نصب نمود. بدین منظور از دستور زیر کمک خواهیم گرفت :

```
$ sudo apt-get install nfs-common portmap
```

۵-٦- ساخت پوشه ی مشترک در کلاینت ها

اکنون نیاز است پوشه ای را بر روی کلاینت ها ایجاد کنیم که حکم دریافت کننده ی فایل از سرور NFS را دارد. یعنی هر فایلی که بر روی سرور نوشته شود بر روی این پوشه کپی خواهد شد. باید توجه داشت که در اینجا نیاز است نام انتخابی برای این پوشه، مشابه نامی باشد که در مرحله ی ۴-۳ انتخاب کرده ایم. پس با استفاده از دستور زیر به ساخت پوشه ی nfsshare/ بر روی کامپیوترهای برده یا همان کلاینت ها می پردازیم :

```
$ sudo mkdir -p /nfsshare
```

۵-۷- راه اندازی پروتکل

وقت آن رسیده تا سرویس مربوط به پروتکل NFS را بر روی سرور راه اندازی کنیم. بدین منظور کافیست دستور زیر را اجرا نماییم :

```
$ sudo /etc/init.d/nfs-kernel-server start
```

۵-۸- سوار کردن[۲۵] پوشه ی سرور بر کلاینت

اکنون نیاز است تا پوشه ی ساخته شده در سرور را بر روی پوشه ی موجود در کلاینت، اصطلاحاً سوار کنیم. برای این کار از دستور زیر کمک خواهیم گرفت و آن را بر روی تمام کامپیوترهای برده اجرا خواهیم کرد :

```
$ sudo mount master:/nfsshare /nfsshare
```

با این کار، زین پس محتویات پوشه ی nfsshare در تمام سیستم ها یکسان خواهد بود. [۱۲]

[۲۵] Mount

فصل ششم
پیاده سازی MPICH

٦-١- مقدمه

همانطورکه پیش تر گفته شد، دریافت و نصب MPICH بر روی پوشه ای انجام خواهد گرفت که از طریق NFS در دسترس تمام سیستم ها می باشد. بنابراین تمام مراحلی که در ادامه ی این فصل شرح داده خواهد شد صرفاً بر روی کامپیوتر مستر یا همان سرور انجام می گیرد.

٦-٢- دریافت پکیج MPICH

اول از هرچیز از آنجایی که قصد داریم پکیج MPICH را به صورت دستی و در پوشه ی موردنظر خودمان نصب نماییم، نیاز است تا فایل فشرده ی آن را از وبسایت رسمی دریافت نماییم. بنابراین با استفاده از دستور زیر، جدیدترین نسخه ی این پکیج را بر روی پوشه ی nfsshare دانلود خواهیم کرد :

```
/nfsshare$ sudo wget http://www.mpich.org/static/downloads/
                    3.2.1/mpich-3.2.1.tar.gz
```

پس از اتمام دانلود می توانیم با دستور زیر به خارج سازی آن از حالت فشرده بپردازیم.

```
/nfsshare$ sudo tar xvf mpich-3.2.1.tar.gz
```

٦-٣- پیکربندی و نصب

اکنون وارد پوشه ی mpich-3.2.1 شده و دستور زیر را جهت اعمال پسوند وارد می کنیم :

```
/nfsshare/mpich-3.2.1$ sudo ./configure --prefix=/mirror/mpich2
```

در صورت بروز مشکل، می توان با دستور زیر پکیج build-essential را نیز بر روی سیستم نصب نمود.

```
$ sudo apt-get install build-essential
```

همچنین امکان بروز مشکل در ارتباط با برخی کامپایلرها وجود خواهد داشت که در اینصورت می توان با اضافه کردن عبارت زیر به دستور قبلی به راحتی از آنها صرف نظر کرد.

```
/nfsshare/mpich-3.2.1$ sudo ./configure --prefix=/mirror/mpich2
                        disable-f77 –disable-fc
```

حال جهت تکمیل نصب، به ترتیب دستور زیر را اجرا کرده و منتظر می مانیم تا تکمیل شود.

```
/nfsshare/mpich-3.2.1$ sudo make
```

```
/nfsshare/mpich-3.2.1$ sudo make install
```

٦-٤- تعریف محیط اجرا

از آن رو که پکیج خود را در یک پوشه ی دلخواه نصب کرده ایم نیاز است تا آن را به محیط های اجرای پیش فرض سیستم عامل اضافه کنیم. بدین منظور به ترتیب ابتدا با دستور زیر مسیر آن را پیکربندی خواهیم کرد.

```
/nfsshare/mpich-3.2.1$ export PATH=/mirror/mpich2/bin:$PATH
```

```
/nfsshare/mpich-3.2.1$ export LD_LIBRARY_PATH=
                        "mirror/mpich2/lib:$LD_LIBRARY_PATH"
```

اکنون آدرس آن را به فایل environment سیستم عامل اضافه خواهیم کرد. ابتدا این فایل را درون ویرایشگر باز کرده :

```
/nfsshare$ sudo pico /etc/environment
```

سپس آدرس زیر را به ابتدای مسیر PATH اضافه خواهیم کرد، یعنی خواهیم داشت :

```
PATH="/nfsshare/mpich2/bin:/usr/local/…"
```

فایل را ذخیره کرده و خارج می شویم. [۱۲]

فصل هفتم
روش اجرای برنامه توسط MPICH

۷-۱- مقدمه

جهت اجرای صحیح یک برنامه توسط MPICH ابتدا بهتر است از صحت عملکرد تمام پیش نیازها نظیر شبکه، پروتکل SSH و پروتکل NFS اطمینان حاصل کرده و همچنین مطمئن شویم که فایروال تمام سیستم ها غیرفعال است. پس از آن جهت اینکه قادر به مشاهده ی عملکرد سیستم و کارکرد پردازنده ها باشیم به نصب پکیج htop بر روی تمام کامپیوترها می پردازیم. سپس همه چیز آماده است تا برنامه های خود را با استفاده از MPICH به صورت موازی بر روی تمام سیستم ها اجرا نماییم.

۷-۲- غیرفعالسازی فایروال

جهت غیرفعالسازی فایروال به منظور جلوگیری از مشکلات احتمالی ارتباطی کافیست از دستور زیر استفاده کرده و آن را بر روی تمام کامپیوترهای مورد آزمایش اجرا کنیم :

```
$ sudo ufw disable
```

۷-۳- نصب htop

برای نصب این پکیج که ما را درباره جزئیات سخت افزاری سیستم، فرآیندهای در حال اجرا، میزان کارکرد پردازنده ها در لحظه و... مطلع می کند کافیست از دستور زیر استفاده نماییم:

```
$ sudo apt-get install htop
```

برای مشاهده ی صفحه ی htop نیز کافیست همین عبارت را در ترمینال وارد نماییم.

۷-۳- ساخت فایل hosts

جهت اجرای برنامه توسط MPICH نیاز است تا یک فایل به نام hosts ایجاد کرده و در آن آدرس آی پی سیستم هایی که قرار است درگیر اجرای برنامه موازی ما شوند را وارد نماییم. همچنین قادر خواهیم بود جهت عملکرد بهتر، تعداد پردازنده های هر کامپیوتر را نیز در جلوی آدرس آی پی آن بنویسیم. پس بدین منظور به داخل پوشه ی /etc/nfsshare رفته و یک فایل جدید به نام hosts خواهیم ساخت. سپس در این فایل به فرم زیر به تعریف سیستم ها می پردازیم :

```
192.168.1.10:2
192.168.1.11:2
192.168.1.12:2
192.168.1.13:4
192.168.1.14:4
```

لازم به ذکر است که عدد نوشته شده بعد از علامت دو نقطه، نشانگر تعداد پردازنده های آن سیستم می باشد.

۷-۵- اجرای برنامه

اکنون که همه چیز آماده است می توانیم نسبت به اجرای برنامه های موازی خود اقدام کنیم. بدین منظور ابتدا فایل برنامه خود را در پوشه ی nfsshare کپی کرده و آن را با استفاده از کامپایلر MPI متناسب با زبان برنامه، کامپایل می کنیم. به عنوان مثال برای کامپایل کردن یک برنامه زبان C از دستور زیر کمک خواهیم گرفت :

```
/nfsshare$ mpicc -o mpi_sample mpi_sample.c
```

حال می توان به اجرای برنامه ی کامپایل شده توسط MPICH پرداخت. بدین منظور به اجرای دستور زیر می پردازیم :

```
/nfsshare$ mpirun -f hosts -n 12 ./mpi_sample
```

بجای عدد ۱۲ می توان تعداد پردازنده هایی که قصد دارید توسط برنامه درگیر شوند را وارد نمود. همچنین بجای عبارت mpi_sample نیز نام برنامه ی خود را وارد خواهیم کرد.

پس از این کار برنامه به خوبی بر روی تمام سیستم ها اجرا گشته و می توان به کمک htop نسبت به مشاهده ی عملکرد پردازنده ها بر روی هر کدام از کامپیوترها اقدام کرد.

پیوست ۱

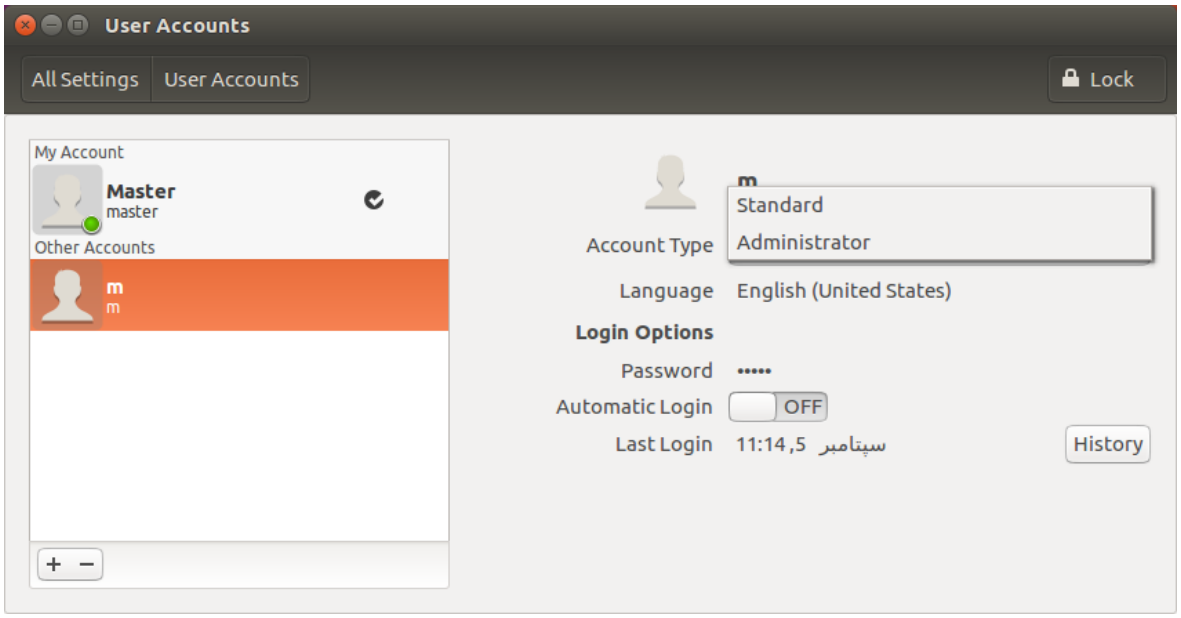
تعریف کاربر جدید در اوبونتو

جهت تعریف یک کاربر جدید در سیستم عامل اوبونتو کافیست ترمینال را باز کرده و دستور زیر را وارد نمود :

```
$ sudo adduser username
```

کافیست بجای عبارت *username*، نام کاربری دلخواه خود را وارد کرده و پس از اجرای دستور بالا یک رمزعبور برای آن انتخاب کنیم. می توان فیلدهای اطلاعاتی بیشتری نیز برای کاربر خود تکمیل نمود که اجباری در این کار نیست.

باید توجه داشت که بصورت پیش فرض به کاربر جدیداً تعریف شده دسترسی مدیریت اعطا نخواهد شد. بدین منظور نیز کافیست وارد بخش User Accounts اوبونتو شده، بر روی دکمه ی Lock در بالای صفحه کلیک کرده و رمز عبور کاربر اصلی سیستم را وارد نماییم. سپس بر روی نام کاربری که اخیراً تعریف کرده ایم کلیک کرده و در منوی کشویی Account Type، حالت آن را به Administrator تغییر دهیم.



شکل (پ۱-۱)

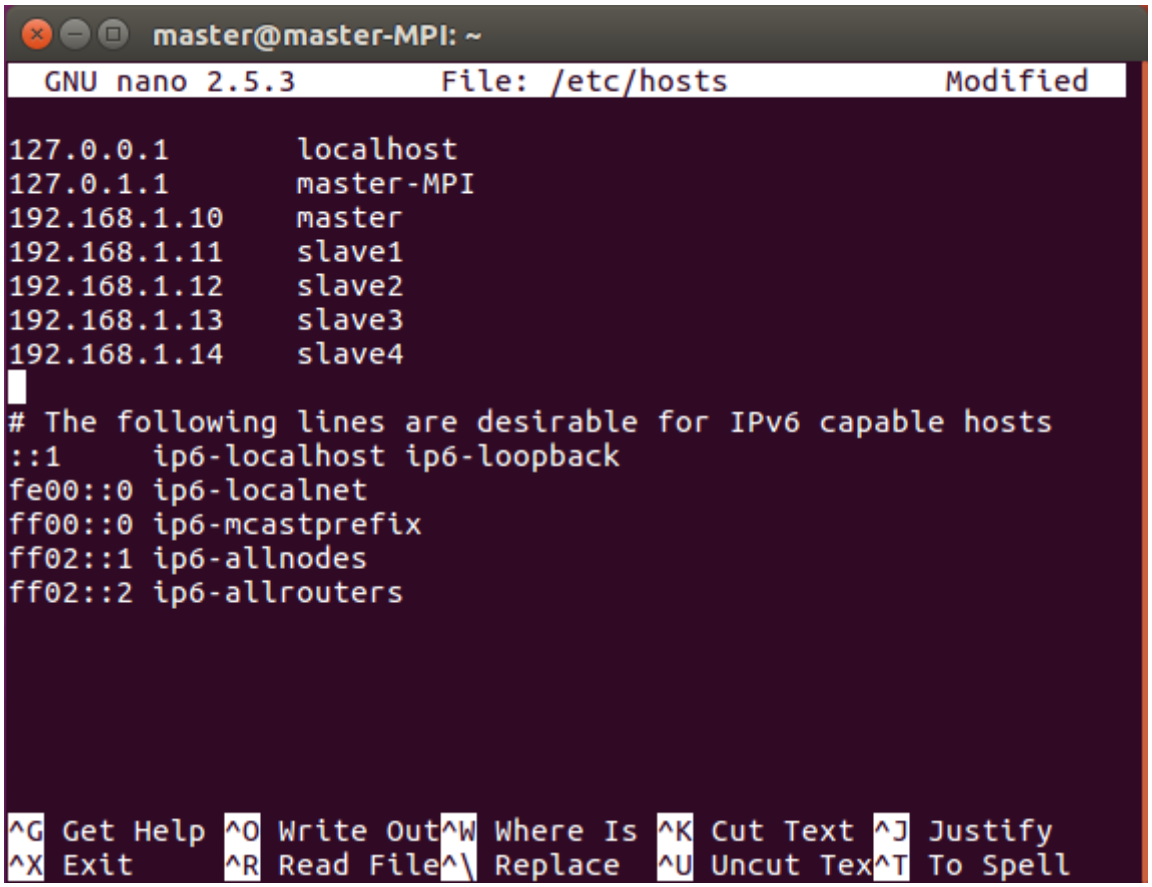
پیوست ۲

تعریف نام میزبان آی پی در اوبونتو

برای تعریف نام میزبان دلخواه برای آی پی ها در سیستم عامل اوبونتو کافیست به پیکربندی فایل hosts در پوشه ی etc بپردازیم. جهت اجرا و امکان ویرایش این فایل می توان با دستور زیر از محیط pico کمک گرفت:

```
$ sudo pico /etc/hosts
```

اکنون می توان به صورت یک لیست، آدرس آی پی هر کدام از کامپیوترها و سپس نام دلخواهی که برای آن بر می گزینیم را نوشت. یعنی در مثال خود وقتی ۵ کامپیوتر در اختیار داریم نیاز است تا در ۵ خط، آی پی هرکدام و نام دلخواهی برای هرکدام از آنها را در این فایل تعریف کنیم. همین کار را بر روی تمام سیستم ها تکرار کرده و بهتر است که نام انتخابی ما بر روی همه ی آنها نیز مشابه باشد. بعد از اتمام کار، کافیست فایل مربوطه را ذخیره نماییم.



شکل (پ۲-۱)

مراجع

[۱] سایت رسمی دانشگاه گیلان، درباره دانشگاه – http://www.guilan.ac.ir

[۲] سایت رسمی دانشگاه گیلان، دانشکده ها، دانشکده فنی – http://www.guilan.ac.ir/engineering

[۳] سایت گروه مهندسی کامپیوتر دانشگاه گیلان – http://www.ce.guilan.ac.ir

[۴] High-performance computing (HPC), Margaret Rouse -
https://searchdatacenter.techtarget.com/definition/high-performance-computing-HPC

[۵] High-Performance Computing (HPC) -
https://www.techopedia.com/definition/4595/high-performance-computing-hpc

[۶] MPI Tutorial Introduction - http://mpitutorial.com/tutorials/mpi-introduction/

[۷] MPICH Overview - https://www.mpich.org/about/overview/

[۸] Secure Shell - https://en.wikipedia.org/wiki/Secure_Shell

[۹] Network File System - https://en.wikipedia.org/wiki/Network_File_System

[۱۰] MPI Matrix Multiply – C version
https://computing.llnl.gov/tutorials/mpi/samples/C/mpi_mm.c

[۱۱] UBUNTU and static network address –
https://www.youtube.com/watch?v=g07VGfILpWw

[۱۲] Distributed and Parallel Computing by Eugene Ch'ng –
https://www.youtube.com/watch?v=2rpWEZY0aPo&list=PLbx-k3N9Yr99WQVwlIdbWdsVxi5ItGL14