راه اندازی یک سیستم پردازش موازی روی لینوکس

آنچه در نهایت خواهید داشت یک سیستم پردازشی موازی با معماری multicomputer است که با کتابخانه mpi (برنامه نویسی موازی به شیوه انتقال پیام) با آن می توانید برنامه بنویسید. شاید برای کسانی که یه تعداد کامپیوتر بی کار در دسترسشون هست و ی مقداری هم وقت دارند که نمی دونن چطور تلف کنند این مقاله خوبی باشه. شاید هم بعدا به درد یه کسی بخوره.

سخت افزار مورد نیاز:
یک عدد رایانه سرور کامل حداقل پنتیوم 3، و حافظه اصلی 256 به بالا، فضای دیسک سخت زیاد (مثلا 10 الی 15 گیگابایت) و یک یا تعداد بیشتری کارت شبکه نسبتا سریع و مادربرد با قابلیت بوت از روی کارت شبکه (معروف به PXE دار).
تعدادی client حداقل پنتیوم 2، حافظه اصلی 128 به بالا) و یک کارت شبکه نسبتا سریع و مادربرد با قابلیت بوت از روی کارت شبکه (معروف به PXE دار) داشتن دیسک سخت و صفحه کلید و موشواره اختیاری است.
در صورت نیاز به switch باید از نوع سریع باشند، HUB های معمولی جواب خوبی نمی دهند.

نرم افزار مورد نیاز:
پیش از شروع کار یک عدد لینوکس (ترجیحا FC3 یا Mandrake یا Debian اگر از روش دوم استفاده می کنید) روی سرور خود نصب کنید. چون روش کار بدین ترتیب است که client ها به طور کامل از روی شبکه بوت می شوند بنابراین حتی ممکن است دیسک سخت هم نداشته باشند. تنها کاری که برای client ها باید انجام دهید این است که در setup آن اولویت بوت را کارت شبکه بگذارید و مطمئن شوید که به یکی از کارت های شبکه سرور شما مستقیم یا با استفاده از یک switch متصل اند.

من دو روش برای راه اندازی سیستم می نیمم ذکر می کنم: روش اول که روش دستی است و انجام آن سخت، ولی با امکانات خط اینترنت معمول قابل انجام است و روش دوم که با استفاده از یک package از قبل نوشته شده است و بسیار آسان می باشد ولی احتمالا خط اینترنت پر سرعت برای نصب لازم دارد.

روش اول (روش دستی):
ابتدا مراحل بوت شدن یک سیستم از روی کارت شبکه را مختصرا ذکر می کنم. وقتی یه سیستم از روی یک کارت شبکه مجهز به seprom که مثلا PXE را ساپورت می کند می خواد بالا بیاد، اول به دنبال یه سرویس BOOTP یا DHCP می گردد تا از آن ip بگیرد. بنابراین باید روی سرور خود یک عدد DHCP یا BOOTP پیکره بندی شده داشته باشید. سپس با استفاده از پروتکلی مثل TFTP (مشابه همون FTP خودمون) کرنل که یه کم هم دستکاری شده دانلود شده و اجرا می شود پس TFTP هم باید سرویسش نصب باشه. بعد احتمالا لازم دارید که یک root دایرکتوری یا مثلا یک دایرکتوری مشابه etc و یا چیز های مشابه برای اینکه سرویس های دیگر را run کنید، mount شود. از آن جایی که دیسک سخت ندارید پس باید این کار را با چیزی شبیه NFS انجام دهید که آن را هم نصب و پیکره بندی می کنید (از این قسمت برای سفارشی کردن clientها مورد استفاده قرار می گیرد).
برای انجام دقیق و مرحله به مرحله این روش شما را به HOWTO زیر راهنمایی می کنم:
/http://www.tldp.org/HOWTO/Network-boot-HOWTO

مراحل کار به صورت خلاصه عبارتند از:
ساخت مجدد کرنل و دستکاری آن با پارامتر های مورد نیاز

نصب و پیکره بندی NFS
نصب و پیکره بندی BOOTP
نصب و پیکره بندی TFTP
تنظیم client ها
حال کردن با سیستمی که راه انداخته اید

روش دوم(روشی دیگر):
یه دانشگاه یا موسسه تایوانی می خواسته یه سیستم راه بندازه که کامپیوترهای شبکه اش بدون هارد لینوکس رو بالا بیارند. واسه همین هم کارهای بالا رو اومده یه کم جمع و جور کرده و ساده تر کرده. کاری که انجام داده اینه که اومده یه سری script به زبان perl نوشته که با اجرای اون ها سیستم نصب و پیکره بندی می شه. آدرس:
/http://drbl.sourceforge.net

با اجرای یکی از برنامه هاش (فکر کنم drblsrv) فرض می کنه که eth0 کامپیوتر سرور به اینترنت وصل هست و بعد شروع می کنه تمام بسته های مورد نیاز و حتی کرنل رو نصب و به روز می کنه و بعد دایرکتوری های معادل client ها رو می سازه.
با اجرای یکی دیگه از برنامه ها (به اسم drblpush) بقیه کارت های شبکه تون (eth1 به بعد) رو به عنوان زیر شبکه هایی در نظر می گیره که هر کدوم یه تعدادی client بهش وصل هستند. Ip ها، تعداد client ها، امکانات هر client و چیزهایی از این قبیل همین جا قابل تنظیم هستند.
بعد از نصب هم یک سری دستورات shell برای تنظیمات آتی در اختیار خواهید داشت. این روش با اینکه خاص هست و برای همه کاربرد ها در نظر گرفته نشده ولی امکانات پیکره بندی ساده و زیادی به شما می ده (مثلا اینکه clientهاتون گرافیکی login کنند یانه، چه جور loginای براشون در نظر گرفته بشه و...). اون سیستمی که من باهاش کار می کردم یه رک بود که سه تا مادربرد تو یه طبقه داشت و طبقه زیریش سه تا powerشون بود و از بیرون به یکیشون کی برد و مانیتور وصل شده بود بنابراین زیاد با امکانات گرافیکی loginاش ور نرفتم ماکزیمم چیزی که می خواستم یه سرور ssh روی دوتای دیگه بود که بتونم با شبکه بهشون دسترسی داشته باشم. عکس هاشو پیوست می کنم.

البته خیال کسی رو بر نداره که الآن می تونیم یه supercomputer بسازیم فقط پولش باید باشه. هر چند که راهش همینه ولی اهل فن (مطمئنا منظورم نرم افزاری ها نیست، شاید سخت افزاری ها که یه کم از رشته ما رو می خونن) می دونن که اون ابر کامپیوترهایی که تو اخبار می گن آرایه های پیچیده ای از پردازنده های طراحی شده برای این کار، ram های switching matrix، multiport های پیچیده و... هست. ضمنا این رو هم حتما می دونید که فعلا تو سرعت تقریبا به بن بست رسیده ایم. چون برای با لا بردن سرعت باید ماسفت ها رو کوچیک کرد ولی تکنولوژی کوانتم مکانیک فعلی جواب کوچیک تر از چند ده نانومتر رو نمی ده به همین خاطر دارن روی پردازنده های چند هسته ای یا سیستم های چند پردازنده ای کار می کنن. البته هر چند که ما نه شبیه سازی انفجار اتمی می کنیم و ماهواره هواشناسی داریم که بخواهیم داده های عظیمشو تحلیل کنیم ولی اگر بخواهیم روزی به این سمت بریم برنامه نویسی موازی به نظر می رسه فعلا تنها راه حله
یک interface ساده و ارزان و در دسترس برای ارتباط پردازنده ها استفاده از یک شبکه ip با تجهیزات مربوطه است. در قسمت قبل مختصرا توضیح دادم که چطور می توان یک cluster بسیار ابتدایی بر پایه لینوکس داشت. قدم بعد از مهیا کردن سخت افزار نوشتن نرم افزاری است که به طور موازی روی این سیستم اجرا شود.
با توجه به معماری مالتی کامپیوتر سیستم ما، به عنوان اولین راه که به ذهن می رسد می توان از توابع شبکه موجود در کتابخانه های زبان های برنامه نویسی معمول که به عنوان مثال می توانید با آن ها یک socket بر پایه tcp درست کنید و قبل و بعد از اجرای هر بخش الگوریتم روی هر پردازنده داده ها

را بین پردازنده ها جا به جا کنید استفاده کرد. این یک کار low-level است و انجام آن برای برنامه های بزرگ نسبتا سخت می باشد. به علاوه ملاحظاتی که در برنامه موازی باید در نظر داشته باشید را ممکن است رعایت نکنید و گذشته از همه این ها کد شما کاملا وابسته به سیستمی است که طراحی کردهاید و قابل انتقال روی یک سیستم دیگر نمی باشد.
در اوایل دهه نود میلادی گروهی با عنوان mpi forum مسئول بررسی کتابخانه های برنامه سازی موازی به شیوه انتقال پیام شدند و با در نظر گرفتن ویژگی های یک برنامه موازی و نقاط ضعف و قوت آن ها کتابخانه توابع mpi را طراحی پیشنهاد کردند. یکی از ویژگی های این کتابخانه این بود که کد نوشته شده توسط شما مستقل از platform بود و شما می توانستید کد خود را روی هر سیستمی که کتابخانه mpi روی آن پیاده سازی شده بود به کار ببرید.
کتابخانه توابع mpi را به طور ساده شاید بتوان مجموعه ای از دستورات که صرفا عمل انتقال بلوک هایی از داده ها از یک node یا گره به گره دیگر را انجام می دهند در نظر گرفت (هر node را یک cluster در نظر بگیرید که می تواند به طور ساده یک مادربرد باشد)
یکی از پیاده سازی های معروف mpi که همراه بسیاری از distro های لینوکس (مانند Fedora) می باشد lam-mpi است. من پورت مربوط به bsd آن را در سایت freebsd.org نیز دیده ام. بعد از نصب این برنامه آنچه شما خواهید داشت کتابخانه ای از توابع مورد استفاده در زبان های c و c++ و FORTRAN است که شما در برنامه نویسی از آن ها استفاده می کنید، یک سرویس به صورت daemon است که هنگام اجرای برنامه ها باید فعال باشد و در نهایت یک سری دستور خط فرمانی برای راه اندازی و توقف سرویس، اجرای برنامه ها و آسان تر کردن عمل compile برنامه ها.
توجه کنید که برای کار با این library لزومی ندارد که سیستمی مانند آنچه در قسمت اول توضیح دادم داشته باشید بلکه اگر فقط دو pc به همراه لینوکس داشته باشید که به هم شبکه شده باشند کفایت می کند.
قدم اول در اجرای الگوریتم به طور موازی نوشتن برنامه است. در خصوص نوشتن برنامه بعدا توضیحاتی می دهم ولی فعلا فرض کنید برنامه نوشته شده و آماده کامپایل و اجرا می باشد. مانند تمامی دیگر برنامه ها باید gcc (یا g++) را به همراه سوئیچ های مربوطه صدا کنید ولی برای این کار شاید مجبور باشید یک دستور دو سه خطی بنویسید زیرا احتمالا مجبورید تعداد زیادی library را به همراه سوئیچ های مربوطه مشخص کنید. چون این آرگومان های خط فرمانی عموما ثابت هستند به همین خاطر دستورهای mpicc و mpiCC (برای cو c++) در نظر گرفته شده که خود به خود کامپایلر را با پارامترهای لازم صدا می کنند که در نتیجه کار ساده می شود.
بعد از کامپایل برنامه و پیش از اجرای آن شما باید محیط lam را فعال کنید تا بتوانید برنامه mpi خود را در آن اجرا کنید. برای این کار ابتدا یک فایل متنی که حاوی آدرس node هایی است که شما قصد دارید برنامه روی آن ها اجرا شود بسازید (در هر node باید lam-mpi و rshیاssh و همچنین یک کپی از برنامه کامپایل شده موجود باشد) سپس با دستور lamboot و مشخص کردن hostfile گفته شده محیط را اجرا می کنید. در نهایت با استفاده از دستورات mpirun یا mpiexec و مشخص کردن اینکه برنامه تان را می خواهید با چند process موازی اجرا نمایید، کار را تمام می کنید.

این مقاله کوتاه صرفا برای آشنایی و درک شمای کلی یک سیستم پردازش موازی نوشته شده است. برای انجام کار جدی می بایست مستندات mpi و همچنین lam-mpi را به دقت مطالعه نمایید.

بستن این پنجره