برنامه‌سازی شبکه

ویژه دانشجویان دانشگاه علمی کاربردی

هادی جلالیان

ایراهستان

۱۳۹۹

سرشناسه:	جلالیان، هادی،۱۳۶۱-
عنوان و نام پدیدآور:	برنامه‌سازی شبکه/ نویسنده هادی جلالیان.
مشخصات نشر:	لامرد: نشر ایراهستان، ۱۳۹۹.
مشخصات ظاهری:	۸۹ص.
شابک:	۹۷۸-۶۰۰-۷۶۸۰-۴۱-۴
وضعیت فهرست نویسی:	فیپا
یادداشت:	کتابنامه: ص.۸۹.
رده‌بندی کنگره:	TK۵۱۰۵/۵
رده‌بندی دیویی:	۰۰۴/۶۲
شماره کتابشناسی ملی:	۷۵۲۱۴۵۰

فارس- لامرد- علامرودشت- کدپستی ۷۴۴۴۱۶۱۶۴۷
irahestanpub@gmail.com -۰۹۱۷۳۸۲۷۰۰۲

انتشارات ایراهستان

برنامه‌سازی شبکه

هادی جلالیان

انتشارات: ایراهستان

قیمت: ۲۲۰۰۰ تومان

شمارگان: ۵۰۰ جلد

شابک: ٤-۴۱-۷٦۸۰-٦۰۰-۹۷۸

فهرست مطالب

مقدمه

با بررسی ساختار پروتکلهای TCP و IP ، طریقه آدرس دهی ماشینها وپروسه های روی هر ماشین را بوسیله آدرس IP و آدرس پورت آموختیم. معمولاً پیاده‌سازی این پروتکلها توسط طراح هر سیستم عامل انجام و بعنوان جزئی از سیستم عامل همراه آن ارائه و نصب می‌شود. در سیستم عامل هایی مثل یونیکس یا MS-Windows که اصل برنامه آن در دسترس نیست این انعطاف وجود ندارد که بتوان در محیطهای آزمایشگاهی برنامه های TCP و IP یا پروتکلهای مرتبط با آنها را تغییراتی داد و نتیجه تغییرات را بررسی و تحلیل کرد لذا نظر علاقمندان به این مورد را به سیستم عامل " لینوکس " جلب می‌نمائیم.

حال فرض میکنیم یک برنامه نویس بخواهد در یک محیط برنامه نویسی مثل C بگونه ای برنامه نویسی کند که محتویات یک فایل درون یک کامپیوتر راه دور را تغییر بدهد یا آنرا روی کامپیوتر خودش منتقل نماید یا فرض کنید یک برنامه نویس موظف شده است که یک محیط پست

الکترونیکی خاص و با امکانات ویژه برای بکارگیری در یک محیط اداری طراحی نماید. برای طراحی چنین برنامه هایی که تماماً در لایه چهارم یعنی لایه کاربردی تعریف می‌شود برنامه نویس باید به نحـوی بـا مفـاهیم برنامه نویسی تحت شبکه آشنا باشد.

در این کتاب اصول کلی برنامه نویسی شبکه و مفهوم "سوکت"[2] را مورد بررسی قرار می‌دهیم و بـا مثالهای ساده روش نوشتن برنامه های کاربردی تحت پروتکل TCP/IP را تشریح خواهیم کـرد. بـرای سادگی کار و همچنین ارائه دید عمیق ، کدهائی که در این کتاب ارائه می‌شوند به زبان C هستند که در محیط سیستم عامل لینوکس و با مترجم **gcc** به زبان ماشین ترجمه شده اند.در حقیقت این کتاب نقطه آغازی است برای تمام برنامه نویسانی که به نحوی مجبور خواهند شـد برنامه کاربردی تحت شبکه اینترنت بنویسند.سنگ بنای تمام برنامه های کاربردی لایه چهارم مفهومی بنام سوکت است که ایـن مفـهوم توسـط طراحان سیستم عامل یونیکس به زیبایی به منظور برقراری ارتباط برنامـه هـای تحـت شـبکه و تبـادل جریان داده بین پروسه ها ابداع شد و در این کتاب باید مفهوم آنرا کالبد شکافی کنیم.

شاید شما این جمله را شنیده باشید "**در دنیای یونیکس هر چیزی میتواند بصورت یـک فـایل تلقی و مدل شود**". تمام عوامل و انواع ورودی و خروجیها (I/O) میتواند توسط سیسـتم فـایل مـدل شود. مثلاً چاپگر میتواند یک فایل باشد (مثلاً فایلی با نام PRN). حال وقتی سیستم عامل چـاپگـر را بصورت یک فایل استاندارد مدل کرده باشد شما با مفاهیمی که از فایلها و چگونگی بکارگیری آنها در محیط برنامه نویسی آموخته اید، برای راه

اندازی چاپگر و چاپ یک متن، میتوانید در برنامه خود عملیات ساده و در عین حال استاندارد زیر را انجام بدهید:

الف) چاپگر را همانند یک فایل با نام استاندارد آن بصورت فایلی "فقط نوشتنی" باز می کنید. (با دستورات (open() یا (fopen()).

ب) اگر نتیجه مرحله قبل موفقیت آمیز بود سیستم عامل یک مشخصه فایل[3] بعنوان اشاره گر فایل برمیگرداند.

ج)دادههائی که قرار است بر روی چاپگر ارسال شوند را با همان دستورات معمولی نوشتن درفایل (دستور معمولی (write() یا fwrite()[2]) ، درون فایل باز شده از مرحله قبل مینویسید.

د)پس از اتمام کار فایل را میبندید. (دستور (close() یا (fclose())

کلیت کاری که باید انجام بشود همین چند مرحله است و برنامه نویس به هیچ عنوان درگیر ساختار چاپگر و اعمالی که برای راه اندازی و چاپ یک متن لازم است نخواهد شد. این وظائف را راه انداز چاپگر بعنوان بخشی از پوسته سیستم عامل بعهده دارد.

چهار مرحلهای که برای بکارگیری چاپگر معرفی شد دقیقا میتواند برای نوشتن بر روی صفحه نمایش یا خواندن از آن مورد استفاده قرار گیرد ، فقط باید نام فایل صفحه نمایش "کنسول" (con)[2] در نظر گرفته شود.

یونیکس قادر است تمام دستگاههای ورودی و خروجی را بعنوان فایل مدل نماید. بن ابراین تمام عملیاتی که برنامه نویس برای بکارگیری دستگاههای

مختلف بایستی بداند و بکار بگیرد یکسان و ساده و شفاف خواهد بود. آیا میتوانید گزاره های زیر را بپذیرید:

- چاپگر فایلی است فقط نوشتنی
- پویشگر[1] فایلی است فقط خواندنی
- صفحه نمایش بعنوان کنسول فایلی است خواندنی و نوشتنی
- پورت سریال فایلی است خواندنی و نوشتنی
- یک فایل واقعی روی دیسک سخت فایلی است خواندنی و نوشتنی
- یک فایل واقعی روی دیسک فشرده فایلی است فقط خواندنی
- صف FIFO[2] یا خط لوله در محیط یونیکس فایلهایی هستند خواندنی و نوشتنی

حال که ذهن شما این نکته را پذیرفت که هر نوع I/O در دنیای سیستم عامل بصورت یک فایل استاندارد قابل عرضه و مدل کردن است ، شما را با یک سوال کلیدی مواجه میکنیم :

" آیا ارتباط دو کامپیوتر روی شبکه و مبادله اطلاعات بین آن دو، ماهیت ورودی / خروجی (I/O) ندارد؟"

اگر جوابتان منفی است این کتاب را رها کنید ولی اگر تردید دارید یا یقیناً جوابتان مثبت است تا انتها این کتاب را دنبال نمایید. اگر ساختار فایل را برای ارتباطات شبکه ای تعمیم بدهیم آنگاه برای برقراری ارتباط بین دو برنامه روی کامپیوترهای راه دور روال زیر پذیرفتنی است :

الف) در برنامه خود از سیستم عامل بخواهید تا شرایط را برای برقراری یک "ارتباط" با کامپیوتری خاص (با آدرس IP مشخص) و برنامه ای خاص روی آن کامپیوتر (با آدرس پورت مشخص) فراهم کند یا اصطلاحاً سوکتی را بگشاید. اگر این کار موفقیت آمیز بود سیستم عامل برای شما یک اشاره گر فایل برمی گرداند و در غیر اینصورت طبق معمول مقدار پوچ (NULL) به برنامه شما برخواهد گرداند.

ب) در صورت موفقیت آمیز بودن عمل در مرحله قبل ، به همان صورتی که درون یک فایل می نویسید یا از آن میخوانید ، میتوانید با توابع send() یا write()و recv()یا read() اقدام به مبادله دادهها بنمائید.

ج) عملیات مبادله دادهها که تمام شد ارتباط را همانند یک فایل معمولی ببندید. (با تابع close()) برای آنکه در برنامه خود همانند فایل یک "اشاره گر ارتباط" را از سیستم عامل طلب کنید تا بتوانید مقدمات یک ارتباط را فراهم کند بایستی تابع سیستمی socket() را در برنامه خود صدا بزنید. درصورتی که عمل موفقیت آمیزبود ، یک اشارهگر غیر پوچ بر میگردد که از آن برای فراخوانی توابع و روالهای بعدی استفاده خواهد شد.

پس از این هرگاه از "سوکت باز" یا مبادله دادهها روی سوکت یاد کردیم منظورمان اشاره به یک ارتباط باز یا مبادله اطلاعات بین دو نقطه TSAP1 روی دو سیستم شبکه کامپیوتری میباشد.

دقیقاً همانند فایلها که میتوان همزمان چندین فایل را در یک برنامه واحد باز کرد و روی هر یک از آنها (با استفاده از اشاره گر فایل) نوشت یا از آنها خواند، در یک برنامه تح ت شبکه میتوان بطور همزمان چندین ارتباط فعال

و باز داشت و با مشخصه هر یک از این ارتباطها روی هر کدام از آنها مبادله داده انجام داد.

انواع سوکت و مفاهیم آنها

اگر بخواهیم از نظر اهمیت انواع سوکت را معرفی کنیم دو نوع سوکت بیشتر وجود ندارد. (انواع دیگری هم هستند ولی کم اهمیت ترند) این دو نوع سوکت عبارتند از :

- سوکتهای نوع استریم که سوکتهای اتصال گرا[2] نامیده میشود.
- سوکتهای نوع دیتاگرام که سوکتهای بدون اتصال[1] نامیده میشود.

اگر عادت به پیش داوری دارید برای تمایز بین مفهوم این دو نوع سوکت، تفاوت بین مفاهیم ارتباط نوع TCP و UDP را مد نظر قرار بدهید. روش ارسال برای سوکتهای نوع استریم همان روش TCP است و بنابراین دادهها با رعایت ترتیب و مطمئن با نظارت کافی برخطاهای احتمالی مبادله میشوند. سوکتهای نوع دیتاگرام نامطمئن است و هیچگونه تضمینی در ترتیب جریان دادهها وجود ندارد. اکثر خدمات و پروتکلهائی که در لایه چهارم تعریف شدهاند و در فصول بعدی آنها را بررسی میکنیم نیازمند حفظ اعتبار و صحت دادهها و همچنین رعایت ترتیب جریان دادهها هستند. بعنوان مثال پروتکل انتقال فایل (FTP)، پروتکل انتقال صفحات ابرمتن (HTTP) یا پروتکل انتقال نامه های الکترونیکی (SMTP) همگی نیازمند برقراری یک ارتباط مطمئن هستند و طبعاً از سوکتهای نوع استریم بهره میبرند.

---

[1] Connection OrientedConnectionless