برنامه نویسی وب، پاییز ۱۴۰۳-استاد: یحیی پورسلطانی

تحقیق: ابزار nginx و کاربرد آن

اعضای تیم:

مهرشاد برزمینی

۹۹۱۷۰۳۶۱

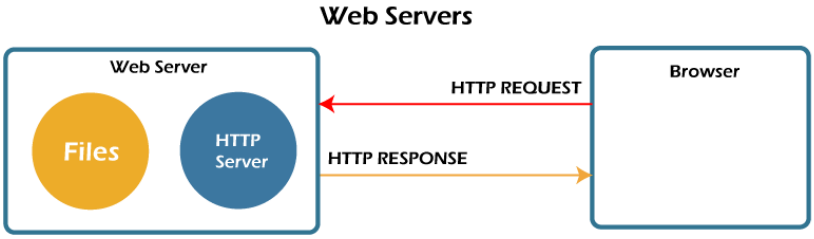
مهدیار احمدی‌زاده

۹۹۱۷۰۳۳۷

فهرست مطالب

۱. سرورهای وب

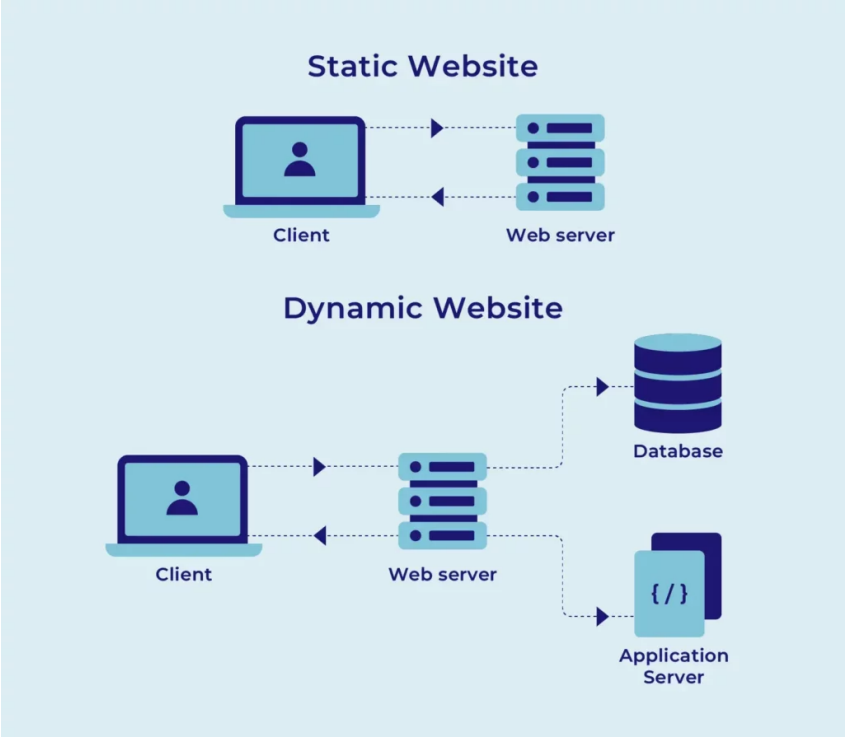
یک **سرور وب** سیستمی است که درخواست‌های مشتریان (معمولاً مرورگرها) را پردازش کرده و محتوای وب مانند صفحات HTML، تصاویر و ویدئوها را ارائه می‌دهد. سرورهای وب اجزای ضروری اینترنت هستند که به کاربران این امکان را می‌دهند که به وب‌سایت‌ها و برنامه‌های وب دسترسی پیدا کنند.



شکل ۱: سرورهای وب

دو نوع اصلی از سرورهای وب وجود دارد:

- **سرورهای وب ایستا**: فایل‌های HTML از پیش نوشته‌شده را بدون تغییرات ارائه می‌دهند.
- **سرورهای وب دینامیک**: درخواست‌ها را پردازش کرده و محتوای دینامیکی با استفاده از منطق پشت‌صحنه تولید می‌کنند.



شکل ۲: وب سرورهای ایستا و پویا

۱.۱ تاریخچه سرورهای وب

سرورهای وب اجزای ضروری اینترنت هستند که امکان ارسال صفحات وب، برنامه‌ها و خدمات را فراهم می‌کنند. تاریخچه سرورهای وب به روزهای اولیه شبکه جهانی وب برمی‌گردد، زمانی که نیاز به ارائه و توزیع محتوای وب به طور مؤثر مشخص شد.

۱.۱.۱ اولین سرور وب

اولین سرور وب توسط **تیم برنرز-لی** در **CERN** در سال **۱۹۹۰** به عنوان بخشی از پروژه شبکه جهانی وب توسعه داده شد. این سرور وب که CERN httpd نام داشت، برای ارائه اسناد ابرمتنی از طریق شبکه طراحی شده بود و پایه‌گذار سرورهای وب مدرن بود که پروتکل HTTP/0.9 را پیاده‌سازی کرد.

۲.۱.۱ تکامل سرورهای وب

با رشد اینترنت، سرورهای وب پیشرفته‌تری برای مدیریت ترافیک زیاد و محتوای دینامیک توسعه یافتند.

۱. **سرور وب Apache HTTP (۱۹۹۵)**: توسعه داده شده توسط **Apache Software Foundation**، Apache تبدیل به محبوب‌ترین سرور وب شد. این سرور **معماری مدولار** را معرفی کرد که از گسترش برای امنیت، کشینگ و تعادل بار پشتیبانی می‌کند. به دلیل پایداری، Apache در اوایل ۲۰۰۰ میلادی غالب بود.

۲. **Microsoft Internet Information Services (IIS) (۱۹۹۶)**: معرفی شده توسط مایکروسافت به عنوان بخشی از Windows NT 4.0. طراحی شده برای **برنامه‌های مبتنی بر ویندوز** با یکپارچگی عمیق در فناوری‌های مایکروسافت.

۳. **Nginx (۲۰۰۴)**: ایجاد شده توسط **ایگور سیسویف** برای حل مشکل C10K (مدیریت بیش از ۱۰،۰۰۰ ارتباط همزمان). از مدل **رویداد-محور و ناهمگام** استفاده می‌کند که آن را بسیار مقیاس‌پذیر می‌کند. این سرور برای وب‌سایت‌های پربازدید و با عملکرد بالا محبوب شد.

۴. **LiteSpeed (۲۰۰۳)**: یک سرور وب تجاری که به عنوان جایگزینی سریع‌تر برای Apache طراحی شده است. از پیکربندی‌های Apache پشتیبانی می‌کند ولی عملکرد و امنیت بهتری دارد.

۵. **Caddy (۲۰۱۵)**: یک سرور وب مدرن با **HTTPS خودکار** که به راحتی می‌توان وب‌سایت‌ها را ایمن و راه‌اندازی کرد. این سرور به زبان Go نوشته شده است و برای سادگی و عملکرد طراحی شده است.

۳.۱.۱ تغییر به سمت ابری و میکروسرویس‌ها

- با افزایش **محاسبات ابری**، سرورهای وب به طور فزاینده‌ای **توزیع‌شده و مقیاس‌پذیر** شده‌اند.
- تکنولوژی‌هایی مانند **Envoy ،NGINX و HAProxy** اکنون به طور گسترده در **محیط‌های کانتینری** (مانند Kubernetes) استفاده می‌شوند.
- پلتفرم‌های بدون سرور، مانند AWS Lambda و Google Cloud Functions، میزبانی وب را بدون نیاز به سرورهای اختصاصی فراهم می‌کنند.

۴.۱.۱ نتیجه‌گیری

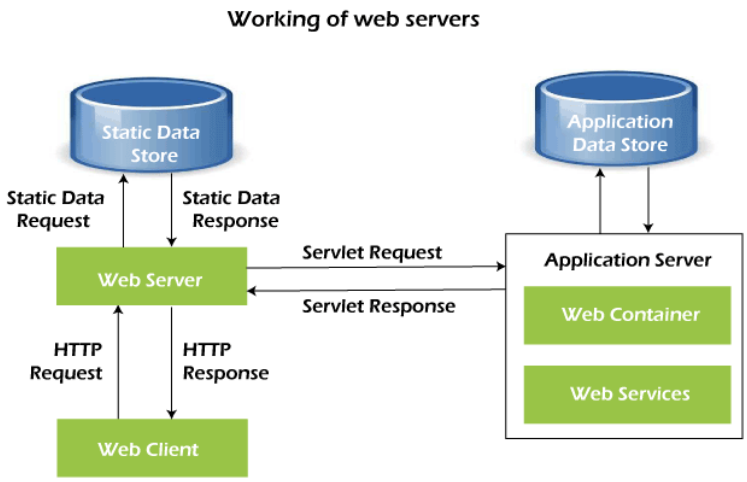
از CERN httpd تا سرورهای وب **مدرن ابری-محور**، سرورهای وب به طور قابل توجهی برای مقابله با نیازهای فزاینده در عملکرد، مقیاس‌پذیری و امنیت تکامل یافته‌اند. امروزه، سرورهای وب نقش اساسی در ارائه تجارب سریع و قابل اعتماد وب ایفا می‌کنند.

۲.۱. نحوه عملکرد سرورهای وب

سرورهای وب از دو جزء اصلی تشکیل شده‌اند: سخت‌افزار و نرم‌افزار. در سطح سخت‌افزاری، سرور وب یک کامپیوتر است که داده‌ها، از جمله فایل‌های HTML، تصاویر و ویدئوها، را ذخیره می‌کند. در سطح نرم‌افزاری، سرور وب شامل نرم‌افزاری است که به مدیریت درخواست‌ها و ارسال پاسخ‌های مناسب می‌پردازد.
فرآیند کلی عملکرد سرور وب به این صورت است:

- مرورگر وب URL را وارد می‌کند.
- مرورگر آدرس IP سرور را از طریق DNS دریافت می‌کند.
- مرورگر درخواست HTTP را به سرور ارسال می‌کند.
- سرور درخواست را پردازش کرده و پاسخ HTTP را با محتوای درخواستی به مرورگر ارسال می‌کند.
- مرورگر صفحه وب را نمایش می‌دهد.

در صورتی که خطایی در درخواست وجود داشته باشد، سرور یک پیام خطا به مرورگر ارسال می‌کند.



شکل ۳: نحوه عملکرد سرورهای وب

۳.۱. امنیت سرورهای وب

برای تضمین امنیت سرورهای وب، تکنیک‌های مختلفی وجود دارند:

- **پروکسی معکوس**: این سرور به عنوان واسطه بین کاربر و سرور اصلی عمل کرده و از آن محافظت می‌کند.
- **محدودیت دسترسی**: با استفاده از **SSH** دسترسی‌های غیرمجاز محدود می‌شود.
- **بروزرسانی‌های منظم**: به‌روزرسانی نرم‌افزار سرور برای جلوگیری از آسیب‌پذیری‌ها ضروری است.
- **فایروال و SSL**: فایروال ترافیک HTTP را نظارت می‌کند و **SSL** داده‌ها را رمزگذاری می‌کند.

۲. مقدمه‌ای بر Nginx

Nginx (که به صورت "engine-x" تلفظ می‌شود) یک سرور وب و سرور پراکسی معکوس با کارایی بالا و متن باز است که برای کارایی، مقیاس‌پذیری و قابلیت اطمینان طراحی شده است. این سرور در ابتدا توسط **ایگور سیسویف** در سال **۲۰۰۴** برای مدیریت اتصالات همزمان زیاد به طور کارآمد توسعه داده شد و امروزه یکی از پرکاربردترین سرورهای وب است، به ویژه برای مدیریت برنامه‌های وب مدرن، میکروسرویس‌ها و وب‌سایت‌های پربازدید.

۱.۲. Nginx چیست؟

Nginx یک سرور وب سبک و مبتنی بر معماری رویداد-محور است که برای ارائه محتوای ایستا، به عنوان یک **پراکسی معکوس**، **تعادل بار** و **سرور کش** عمل می‌کند. برخلاف سرورهای وب سنتی مانند **Apache** که از مدل مبتنی بر فرآیند استفاده می‌کنند، **Nginx** از معماری **ناهمگام و غیرمسدودکننده** پیروی می‌کند که آن را در بارهای سنگین بسیار مقیاس‌پذیرتر و کارآمدتر می‌کند.

۱.۱.۲ ویژگی‌های کلیدی Nginx

- **معماری رویداد-محور**: هزاران اتصال را با کمترین استفاده از منابع مدیریت می‌کند.
- **پراکسی معکوس**: به عنوان واسطه بین مشتریان و سرورهای پشتیبان عمل می‌کند.
- **تعادل بار**: ترافیک را بین سرورهای پشتیبان مختلف توزیع می‌کند.
- **ارائه فایل‌های ایستا**: فایل‌های ایستا مانند HTML، CSS، JavaScript و تصاویر را به طور کارآمد ارائه می‌دهد.
- **ویژگی‌های امنیتی**: از SSL/TLS، کاهش حملات DDoS و مکانیزم‌های احراز هویت پشتیبانی می‌کند.
- **پشتیبانی از FastCGI**: با PHP، Python و دیگر برنامه‌های دینامیک کار می‌کند.

۲.۲. چرا از Nginx استفاده کنیم؟

Nginx به دلایل مختلفی نسبت به سایر سرورها ترجیح داده می‌شود، به ویژه به دلیل **عملکرد بالا، کارایی و معماری مدرن** آن.

۱.۲.۲ مزایای عملکرد

- **مدیریت ترافیک بالا به طور کارآمد:** طراحی شده برای مدیریت ۱۰،۰۰۰+ اتصال همزمان با استفاده کم از حافظه (**حل مشکل C10K**).
- **مدیریت بهینه فایل‌های ایستا:** **Nginx** فایل‌های ایستا را سریع‌تر از **Apache** ارائه می‌دهد به دلیل تعامل بهینه با سیستم فایل.
- **استفاده کمتر از CPU و حافظه:** نسبت به سرورهای مبتنی بر فرآیند مانند **Apache** منابع کمتری استفاده می‌کند، که آن را برای برنامه‌های با عملکرد بالا ایده‌آل می‌سازد.

۲.۲.۲ قابلیت اطمینان و مقیاس‌پذیری

- **معماری غیرمسدودکننده و ناهمگام:** برخلاف مدل مبتنی بر فرآیند **Apache**، **Nginx** درخواست‌ها را بدون ایجاد **رشته‌های جدید** پردازش می‌کند که به آن اجازه می‌دهد به طور مؤثری مقیاس‌پذیر باشد.
- **راه‌اندازی مجدد و بارگذاری پیکربندی بدون وقفه:** تغییرات پیکربندی نیازی به توقف سیستم ندارد.
- **پشتیبانی از پراکسی معکوس و تعادل بار:** ترافیک را به طور مؤثر بین سرورهای پشتیبان توزیع می‌کند و از بروز مشکلات در ترافیک زیاد جلوگیری می‌کند.

۳.۲.۲ مزایای امنیتی

- **حفاظت در برابر DDoS:** با استفاده از محدودیت درخواست‌ها، محدودیت نرخ و محدودیت اتصالات از حملات DDoS جلوگیری می‌کند.
- **ترمینیشن SSL/TLS:** به عنوان **SSL Offloader** عمل می‌کند و رمزنگاری HTTPS را انجام می‌دهد، که بار سرورهای پشتیبان را کاهش می‌دهد.
- **محدود کردن دسترسی با احراز هویت و فایروال‌ها:** از احراز هویت HTTP، سفیدسازی و سیاه‌سازی IP برای جلوگیری از دسترسی غیرمجاز پشتیبانی می‌کند.

۳.۲. مقایسه Nginx با سایر سرورهای وب

Nginx اغلب با **Apache**، **مای‌کروسافت IIS** و **LiteSpeed** مقایسه می‌شود. هر یک از این سرورها ویژگی‌ها و معایب خود را دارند. در جدول زیر مقایسه این سرورها آورده شده است:

ویژگی	Nginx	Apache	IIS	LiteSpeed
معماری	رویداد-محور، غیرمسدودکننده	مبتنی بر فرآیند، مسدودکننده	ترکیبی (رویداد-محور + رشته‌ها)	رویداد-محور، بهینه‌سازی‌شده
عملکرد	همزمانی بالا، حافظه کم	کندتر در بارهای سنگین	بهینه‌شده برای ویندوز	عملکرد بالا، حافظه کم
فایل‌های ایستا	سریع‌ترین	متوسط	کند	سریع‌تر از **Apache**
تعادل بار	پشتیبانی داخلی	نیاز به ماژول‌ها	پشتیبانی محدود	پشتیبانی داخلی
پراکسی معکوس	پشتیبانی داخلی	نیاز به ماژول‌ها	پشتیبانی داخلی	پشتیبانی داخلی
امنیت	بالا (حفاظت در برابر DDoS، SSL)	متوسط (وابسته به ماژول‌ها)	بالا (یکپارچگی با ویندوز)	بالا (پشتیبانی از ModSecurity)
انعطاف‌پذیری	بالا (مدولار)	بسیار بالا (ماژول‌های متعدد)	محدود به ویندوز	بالا (سازگار با **Apache**)

۱.۳.۲ چه زمانی باید از Nginx استفاده کنیم؟

- اگر به **همزمانی بالا** نیاز دارید (مدیریت هزاران درخواست به طور کارآمد).
- اگر به ارائه **فایل‌های ایستا** با استفاده کم از منابع نیاز دارید.
- اگر به **پراکسی معکوس یا تعادل بار** با پشتیبانی داخلی نیاز دارید.
- اگر از معماری **میکروسرویس‌ها** با استفاده از کانتینرها استفاده می‌کنید.

زمانی که Apache یا IIS ممکن است انتخاب بهتری باشند:

- اگر برنامه شما به **فایل‌های htaccess.** (ویژه **Apache**) وابسته است.
- اگر به **یکپارچگی عمیق با Windows Server** نیاز دارید (**IIS** برای این منظور طراحی شده است).
- اگر به **پشتیبانی از ماژول‌های وسیع** نیاز دارید (که **Apache** بیشترین ماژول‌ها را دارد).

۴.۲. استفاده‌های رایج از Nginx

Nginx در حوزه‌های مختلفی استفاده می‌شود، از وب‌سایت‌های شخصی کوچک گرفته تا راه‌حل‌های سازمانی مقیاس‌بالا.

۱.۴.۲ سرور وب برای محتوای ایستا

- **Nginx** به طور کارآمد HTML، CSS، JavaScript، تصاویر و ویدیوها را با حداقل استفاده از CPU و حافظه ارائه می‌دهد.
- ایده‌آل برای **شبکه‌های تحویل محتوا (CDN)** به دلیل توانایی آن در کش کردن و ارائه محتوای ایستا.

۲.۴.۲ پراکسی معکوس برای تعادل بار

- ترافیک را بین سرورهای پشتیبان مختلف توزیع می‌کند تا عملکرد بهبود یابد.
- از الگوریتم‌های مختلف تعادل بار مانند:
  - **Round Robin** (توزیع یکنواخت درخواست‌ها).
  - **Least Connections** (ارسال ترافیک به سرور با کمترین اتصالات).
  - **IP Hash** (اطمینان از اتصال مشتری به همان سرور پشتیبان).

۳.۴.۲ کشینگ برای بهینه‌سازی عملکرد

- **کش پراکسی** سرعت پاسخ‌ها را با ذخیره محتوای درخواست شده به دفعات زیاد افزایش می‌دهد.
- **کش FastCGI** عملکرد **PHP** و برنامه‌های دینامیک را بهینه می‌کند.
- **Microcaching** نسخه‌های کش کوتاه‌مدت را ذخیره می‌کند تا زمان بارگذاری را کاهش دهد.

۴.۴.۲ ترمینیشن SSL و مدیریت HTTPS

- **Nginx** معمولاً برای **آف‌لود کردن رمزنگاری SSL/TLS** برای کاهش بار سرورهای پشتیبان استفاده می‌شود.
- به طور کارآمد **HTTP را به HTTPS** هدایت می‌کند تا امنیت بهبود یابد.

۵.۴.۲ دروازه API برای میکروسرویس‌ها

- درخواست‌های API را به سرویس‌های مناسب هدایت می‌کند و به عنوان یک **دروازه API** عمل می‌کند.
- مفید برای **Kubernetes و محیط‌های مبتنی بر کانتینر**.

۶.۴.۲ پراکسی WebSockettext برای برنامه‌های بلادرنگ

- از **WebSockets** پشتیبانی می‌کند و برنامه‌های بلادرنگ مانند برنامه‌های چت، به‌روزرسانی‌های زنده و خدمات **IoT** را امکان‌پذیر می‌سازد.
- با **Node.js**، **Python** و برنامه‌های مبتنی بر **WebRTC** کار می‌کند.

۷.۴.۲ مدیریت وب‌سایت‌های پربازدید

- توسط **Netflix**، **Airbnb**، **Cloudflare** و **WordPress.com** برای مدیریت میلیون‌ها درخواست در ثانیه استفاده می‌شود.
- به مقیاس‌پذیری **وب‌سایت‌های تجارت الکترونیک، استریم و اخبار** به طور کارآمد کمک می‌کند.

۳. نصب و راه‌اندازی Nginx

Nginx در سیستم‌عامل‌های **Linux** و **Windows** به‌طور گسترده در دسترس است. فرآیند نصب بسته به سیستم‌عامل متفاوت است، اما مراحل نصب ساده هستند. پس از نصب، مدیریت خدمات Nginx، مانند شروع، توقف و راه‌اندازی مجدد، برای حفظ یک محیط پایدار سرور ضروری است.

۱.۳. نصب Nginx در لینوکس

Nginx در مدیر بسته‌های بیشتر توزیع‌های **Linux** موجود است که نصب را ساده می‌کند.

۱.۱.۳ نصب در Debian/Ubuntu

- **به‌روزرسانی فهرست بسته‌ها:** این کار اطمینان می‌دهد که سیستم شما آخرین نسخه‌های بسته‌ها را دریافت می‌کند.
- **نصب Nginx**: نصب آخرین نسخه پایدار.

```
sudo apt update
sudo apt install nginx -y
```

۲.۱.۳ نصب در CentOS/RHEL

- **فعال‌سازی مخزن EPEL**: بسته‌های اضافی، از جمله Nginx را فراهم می‌کند.
- **نصب Nginx**: سرور را دانلود و نصب می‌کند.

```
sudo yum install epel-release -y
sudo yum install nginx -y
```

۳.۱.۳ کامپایل از منبع (برای ساخت‌های سفارشی)

- **دانلود کد منبع Nginx**: آخرین نسخه Nginx را دانلود می‌کند.
- **کامپایل و نصب**: نسخه‌ای سفارشی برای نیازهای خاص فراهم می‌کند.

```
wget http://nginx.org/download/nginx-1.24.0.tar.gz
tar -xvzf nginx-1.24.0.tar.gz
cd nginx-1.24.0
./configure
make
sudo make install
```

۲.۳. نصب Nginx در ویندوز

Nginx به‌طور بومی برای ویندوز بهینه‌سازی نشده است، اما همچنان می‌توان آن را به‌عنوان سرور توسعه یا آزمایشی نصب و اجرا کرد.

۱.۲.۳ دانلود و استخراج Nginx

- **مراجعه به وب‌سایت رسمی**: نسخه Nginx برای ویندوز را از [nginx.org](https://nginx.org/en/download.html) دانلود کنید.
- **استخراج فایل‌ها**: از ابزاری مانند **7-Zip** یا **WinRAR** برای استخراج آرشیو استفاده کنید.

```
Expand-Archive -Path nginx-1.24.0.zip -DestinationPath C:\nginx
```

۲.۲.۳ اجرای Nginx در ویندوز

- **باز کردن خط فرمان**: به دایرکتوری Nginx بروید.
- **شروع سرور**: Nginx را به‌صورت دستی اجرا کنید.

```
cd C:\nginx
start nginx
```

۳.۲.۳ توقف Nginx در ویندوز

- **توقف فرآیند Nginx**: سیگنال خاتمه را ارسال کنید.

```
nginx -s stop
```

۳.۳. شروع، توقف و راه‌اندازی مجدد Nginx

مدیریت Nginx برای اطمینان از عملکرد روان، اعمال تغییرات پیکربندی و رفع مشکلات ضروری است.

۱.۳.۳ مدیریت Nginx در لینوکس

- **شروع Nginx**: درخواست‌های وب را سرو می‌کند.
- **توقف Nginx**: همه عملیات را بلافاصله متوقف می‌کند.
- **راه‌اندازی مجدد Nginx**: Nginx را متوقف و دوباره شروع می‌کند.
- **بارگذاری مجدد پیکربندی**: تغییرات را بدون توقف سرور اعمال می‌کند.

```
sudo systemctl start nginx
sudo systemctl stop nginx
sudo systemctl restart nginx
sudo systemctl reload nginx
```

۲.۳.۳ مدیریت Nginx در ویندوز

- **توقف ایمن Nginx**: اطمینان از خاموش شدن ایمن.
- **بارگذاری مجدد پیکربندی**: اعمال به‌روزرسانی‌ها بدون توقف سیستم.

```
nginx -s quit
nginx -s reload
```

۴.۳. بررسی نصب Nginx

پس از نصب، بررسی این‌که Nginx به‌درستی اجرا می‌شود ضروری است.

۱.۴.۳ بررسی وضعیت Nginx در لینوکس

- **بررسی فعال بودن Nginx**: نمایش می‌دهد که آیا سرور در حال اجرا است.

```
sudo systemctl status nginx
```

- **بررسی پورت‌های شنود**: اطمینان از این‌که Nginx در پورت مورد نظر (**80**) در حال اجرا است.

```
sudo netstat -tulnp | grep nginx
```

۲.۴.۳ بررسی وضعیت Nginx در ویندوز

- **بررسی فرآیندهای در حال اجرا**: تأیید می‌کند که Nginx فعال است.

```
tasklist | findstr nginx
```

۳.۴.۳ آزمایش Nginx از طریق مرورگر وب

- **باز کردن مرورگر وب**: به http://localhost یا آی‌پی سرور خود بروید.
- **خروجی مورد انتظار**: صفحه خوش‌آمدگویی Nginx را نمایش می‌دهد که نشان می‌دهد نصب با موفقیت انجام شده است.

۴. مبانی پیکربندی Nginx

Nginx از یک سیستم پیکربندی ساختاری برای تعریف نحوه پردازش درخواست‌ها، سرو کردن محتوا و مدیریت اتصالات استفاده می‌کند. فایل پیکربندی اصلی، `nginx.conf`، امکان سفارشی‌سازی رفتار سرور از طریق دستورات و ماژول‌ها را فراهم می‌آورد. درک نحوه اصلاح و مدیریت این پیکربندی برای بهینه‌سازی عملکرد، امنیت و قابلیت‌های سرور ضروری است.

۱.۴. nginx.conf چیست؟

فایل `nginx.conf` فایل پیکربندی مرکزی برای Nginx است که معمولاً در مسیر زیر قرار دارد:

- **Debian/Ubuntu**: `/etc/nginx/nginx.conf`
- **CentOS/RHEL**: `/etc/nginx/nginx.conf`
- **Windows**: `C:\ngin\conf\nginx.conf`

۱.۱.۴ ساختار nginx.conf

فایل پیکربندی از اصل‌ی ب‌خش س‌ه تشکیل شده است:

- **بخش عمومی:** شامل تنظیمات عمومی مانند تعداد پردازش‌های کاری.
- **بخش HTTP:** رفتار سرور وب را تعریف می‌کند، شامل بخش‌های سرور و نحوه پردازش درخواست‌ها.
- **بخش سرور:** تنظیمات مربوط به دامنه‌ها یا وب‌سایت‌های فردی.

```
worker_processes auto; stringstyle stringstyle#stringstyle stringstyleAdjusts
    stringstyle stringstylebasedstringstyle stringstyleonstringstyle stringstyleCPU
    stringstyle stringstylecores

events {
    worker_connections 1024; stringstyle stringstyle#stringstyle
        stringstyleLimitsstringstyle stringstyleconcurrentstringstyle
        stringstyleconnectionsstringstyle stringstyleperstringstyle
        stringstyleworker
}

http {
    include mime.types; stringstyle stringstyle#stringstyle stringstyleLoads
        stringstyle stringstylefilestringstyle stringstyletypestringstyle
        stringstylemappings
    default_type application/octet-stream;

    server {
        listen 80;
        server_name example.com;
        root /var/www/html;
        index index.html;
    }
}
```

۲.۱.۴ دستورات مهم در nginx.conf

- `worker_processes`: تعداد پردازش‌های کاری را که درخواست‌ها را پردازش می‌کنند تعریف می‌کند.
- `worker_connections`: بیشترین تعداد اتصالات همزمان برای هر پردازش کاری را تنظیم می‌کند.
- `server_name`: دامنه یا زیر دامنه‌ای را که سرور به آن پاسخ می‌دهد مشخص می‌کند.
- `root`: مسیر ریشه دایرکتوری که Nginx از آن فایل‌ها را سرو می‌کند را تعریف می‌کند.
- `index`: صفحه پیش‌فرض ورودی (مثلاً index.html) را مشخص می‌کند.

۲.۴. بلاک‌های سرور (میزبان‌های مجازی)

بلاک‌های سرور (که به آن‌ها Virtual Hosts نیز گفته می‌شود) به Nginx این امکان را می‌دهند که چندین وب‌سایت یا برنامه را از یک سرور واحد سرو کند.

۱.۲.۴ مثال ساده از بلاک سرور

پیکربندی زیر example.com را با ریشه دایرکتوری در /var/www/example سرو می‌کند:

```
server {
    listen 80;
    server_name example.com www.example.com;
    root /var/www/example;
    index index.html index.htm;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

۲.۲.۴ پیکربندی چند دامنه

Nginx می‌تواند چندین دامنه را با استفاده از بلاک‌های سرور جداگانه سرو کند.

```
server {
    listen 80;
    server_name site1.com;
    root /var/www/site1;
}

server {
    listen 80;
    server_name site2.com;
    root /var/www/site2;
}
```

۳.۲.۴ هدایت HTTP به HTTPS

ترافیک HTTP را به HTTPS برای اتصالات امن هدایت کنید.

```
server {
    listen 80;
    server_name example.com;
    return 301 https://example.com$request_uri;
}
```

۳.۴ مدیریت فایل‌های پیکربندی

به جای اصلاح مستقیم nginx.conf، Nginx پیکربندی مدولار را با قرار دادن تنظیمات خاص سایت در فایل‌های جداگانه فراهم می‌کند.

۱.۳.۴ ساخت فایل‌های پیکربندی خاص سایت

در Debian/Ubuntu، پیکربندی‌ها در /etc/nginx/sites-available/ مدیریت می‌شوند و سپس به /etc/nginx/sites-enabled/ لینک می‌شوند.

```
sudo nano /etc/nginx/sites-available/example.com
```

بلاک سرور را داخل فایل تعریف کنید:

```
server {
    listen 80;
    server_name example.com;
    root /var/www/example;
    index index.html;
}
```

سایت را فعال کنید:

```
sudo ln -s /etc/nginx/sites-available/example.com /etc/nginx/sites-enabled/
```

در CentOS/RHEL، پیکربندی‌ها معمولاً به /etc/nginx/conf.d/ اضافه می‌شوند.

```
sudo nano /etc/nginx/conf.d/example.com.conf
```

۲.۳.۴ غیرفعال کردن سایت

در سیستم‌های مبتنی بر Debian، سایت را با حذف لینک نمادین غیرفعال کنید:

```
sudo rm /etc/nginx/sites-enabled/example.com
```

در سیستم‌های مبتنی بر CentOS، سایت را با تغییر نام فایل غیرفعال کنید:

```
sudo mv /etc/nginx/conf.d/example.com.conf /etc/nginx/conf.d/example.com.
    conf.disabled
```

۴.۴. بارگذاری مجدد پیکربندی به‌صورت ایمن

پس از اعمال تغییرات، Nginx باید برای اعمال تنظیمات جدید بدون وقفه بارگذاری شود.

۱.۴.۴ آزمایش پیکربندی قبل از بارگذاری مجدد

برای جلوگیری از خرابی سرور، فایل‌های پیکربندی را قبل از بارگذاری مجدد آزمایش کنید.

```
sudo nginx -t
```

اگر خطای نحوی وجود داشته باشد، نمایش داده می‌شود.

۲.۴.۴ بارگذاری مجدد Nginx بدون وقفه

برای اعمال تغییرات بدون متوقف کردن Nginx:

```
sudo systemctl reload nginx
```

۳.۴.۴ راه‌اندازی مجدد Nginx (در صورت لزوم)

اگر تغییرات عمده‌ای نیاز به راه‌اندازی مجدد کامل داشته باشند:

```
sudo systemctl restart nginx
```

۵. استفاده از Nginx به عنوان سرور وب

Nginx به طور گسترده‌ای برای سرو کردن محتواهای استاتیک و دینامیک به صورت کارآمد استفاده می‌شود. معماری با عملکرد بالا آن، Nginx را به انتخابی عالی برای میزبانی وب‌سایت‌ها، ها□□□ و برنامه‌های وب تبدیل می‌کند.

۱.۵. سرو کردن فایل‌های استاتیک

Nginx می‌تواند به طور کارآمد محتوای استاتیکی مانند HTML، CSS، JavaScript، تصاویر و ویدیوها را بدون نیاز به پردازش در پشت‌صحنه سرو کند.

۱.۱.۵ پیکربندی پایه برای فایل‌های استاتیک

- `root`: دایرکتوری که فایل‌ها در آن ذخیره شده‌اند را مشخص می‌کند.
- `index`: صفحه پیش‌فرضی که هنگام دسترسی به دایرکتوری سرو می‌شود را تعریف می‌کند.

```
server {
    listen 80;
    server_name example.com;
    root /var/www/html;
    index index.html;
}
```

۲.۱.۵ مدیریت انواع MIME

Nginx به طور خودکار انواع فایل‌ها را با استفاده از `mime.types` تشخیص می‌دهد که در بخش `http` تعریف شده است.

```
http {
    include mime.types;
    default_type application/octet-stream;
}
```

۲.۵. پیکربندی فایل‌های ایندکس

دستور `index` تعیین می‌کند که کدام فایل هنگام دسترسی به یک دایرکتوری سرو می‌شود.

۱.۲.۵ تنظیم یک فایل ایندکس پیش‌فرض

- Nginx از اولین فایل موجود در لیست استفاده می‌کند.

```
server {
    listen 80;
    server_name example.com;
    root /var/www/html;
    index index.html index.htm index.php;
}
```

۲.۲.۵ اجبار به استفاده از یک فایل ایندکس خاص

اگر فایل ایندکس موجود نباشد، رفتار پیش‌فرض بازگشت به خطای ۴۰۳ `Forbidden` است. برای جلوگیری از این موضوع، می‌توان یک صفحه خطای سفارشی تعریف کرد.

```
server {
    listen 80;
    server_name example.com;
    root /var/www/html;
    index index.html;
    error_page 403 = /index.html;
}
```

۳.۵. مدیریت فهرست‌های دایرکتوری

به طور پیش‌فرض، Nginx به دلایل امنیتی اجازه فهرست‌گذاری دایرکتوری را نمی‌دهد. با این حال، در صورت لزوم می‌توان آن را فعال کرد.

۱.۳.۵ فعال کردن فهرست‌گذاری دایرکتوری

- دستور `autoindex on;` فهرستی از فایل‌ها در دایرکتوری نمایش می‌دهد.

```
server {
    listen 80;
    server_name example.com;
    root /var/www/html;

    location /files/ {
        autoindex on;
        autoindex_exact_size off;
        autoindex_format html;
    }
}
```

۲.۳.۵ کنترل قالب‌بندی فهرست دایرکتوری

- autoindex_exact_size off; اندازه فایل‌ها را به صورت KB/MB به جای بایت نمایش می‌دهد.
- autoindex_format html; خروجی را به جای متن ساده، به صورت HTML قالب‌بندی می‌کند.

۴.۵ اجرای برنامه‌های دینامیک (PHP، Python، Node.js)

Nginx می‌تواند محتوای دینامیک را با عمل کردن به عنوان یک پراکسی معکوس یا با استفاده از FastCGI برای پردازش در پشت‌صحنه سرو کند.

۱.۴.۵ اجرای PHP با FastCGI

- PHP به یک مدیر فرآیند FastCGI (PHP-FPM) برای پردازش درخواست‌ها نیاز دارد.

```
server {
    listen 80;
    server_name example.com;
    root /var/www/html;

    location ~ \.php$ {
        include fastcgi_params;
        fastcgi_pass unix:/run/php/php-fpm.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    }
}
```

۲.۴.۵ اجرای برنامه‌های Python (WSGI + Gunicorn)

- Nginx به طور مستقیم Python را اجرا نمی‌کند بلکه درخواست‌ها را به یک سرور WSGI مانند Gunicorn پراکسی می‌کند.

```
server {
    listen 80;
    server_name example.com;

    location / {
        proxy_pass http://127.0.0.1:8000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }
}
```

۳.۴.۵ اجرای برنامه‌های Node.js

- Nginx درخواست‌ها را به یک فرآیند Node.js در حال اجرا پراکسی می‌کند.

```
server {
    listen 80;
    server_name example.com;

    location / {
        proxy_pass http://127.0.0.1:3000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }
}
```

۶. استفاده از Nginx به عنوان پراکسی معکوس

Nginx به طور گسترده‌ای به عنوان یک reverse proxy (پراکسی معکوس) برای ایفا کردن نقش واسطه بین درخواست‌های کاربران و سرورهای پشت‌صحنه استفاده می‌شود. این کار باعث بهبود امنیت، توزیع بار و عملکرد برنامه‌های وب می‌شود.

۱.۶. پراکسی معکوس چیست؟

یک reverse proxy سروری است که بین درخواست‌های کاربران و سرورهای پشت‌صحنه قرار می‌گیرد و درخواست‌ها و پاسخ‌ها را به سرورهای پشت‌صحنه ارسال و دریافت می‌کند. برخلاف یک پراکسی سنتی که درخواست‌های کاربران را به اینترنت ارسال می‌کند، پراکسی معکوس درخواست‌ها را به نمایندگی از برنامه‌های پشت‌صحنه مدیریت می‌کند.

۱.۱.۶ مزایای استفاده از پراکسی معکوس

- **امنیت:** جزئیات سرورهای پشت‌صحنه را مخفی می‌کند و از دسترسی مستقیم کاربران جلوگیری می‌کند.
- **توزیع بار:** ترافیک را بین چندین سرور پشت‌صحنه توزیع می‌کند تا عملکرد بهینه شود.
- **پایان SSL:** رمزنگاری HTTPS را مدیریت می‌کند و بار روی سرورهای پشت‌صحنه را کاهش می‌دهد.
- **کشینگ:** با ذخیره‌سازی محتوای پر بازدید سرعت دسترسی را بهبود می‌بخشد.

۲.۶. پیکربندی پراکسی معکوس پایه

Nginx را می‌توان به راحتی به عنوان یک پراکسی معکوس پیکربندی کرد تا درخواست‌ها را به یک برنامه در حال اجرا در سرور یا پورت دیگر ارسال کند.

۱.۲.۶ پیکربندی ساده پراکسی معکوس

پیکربندی زیر درخواست‌ها را به سرور پشت‌صحنه‌ای که در 5000 پورت در حال اجرا است ارسال می‌کند:

```
server {
    listen 80;
    server_name example.com;

    location / {
        proxy_pass http://127.0.0.1:5000;
    }
}
```

۲.۲.۶ پراکسی معکوس برای چندین سرور پشت‌صحنه

برای مسیریابی ترافیک به سرویس‌های مختلف، می‌توان از چندین بلوک `location` استفاده کرد:

```
server {
    listen 80;
    server_name example.com;

    location /api/ {
        proxy_pass http://127.0.0.1:5001;
    }

    location /static/ {
        proxy_pass http://127.0.0.1:5002;
    }
}
```

۳.۶. ارسال هدرها و درخواست‌ها

هنگامی که Nginx به عنوان پراکسی معکوس عمل می‌کند، باید هدرها را به درستی ارسال کند تا سرورهای پشت‌صحنه اطلاعات صحیحی از درخواست‌های کاربران دریافت کنند.

۱.۳.۶ هدرهای رایج برای ارسال

- `Host`: نام دامنه درخواست اصلی را حفظ می‌کند.
- `X-Real-IP`: آدرس IP واقعی کاربر را ارسال می‌کند.
- `X-Forwarded-For`: آدرس IP کاربر را از چندین پراکسی ردیابی می‌کند.
- `X-Forwarded-Proto`: نشان می‌دهد که درخواست به صورت HTTP یا HTTPS ارسال شده است.

```
server {
    listen 80;
    server_name example.com;

    location / {
        proxy_pass http://127.0.0.1:5000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

۲.۳.۶ مدیریت WebSockets و اتصالات بلندمدت

برای WebSockets، هدرهایی مانند `Upgrade` و `Connection` باید ارسال شوند.

```
server {
    listen 80;
    server_name example.com;

    location /ws/ {
        proxy_pass http://127.0.0.1:5000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "Upgrade";
    }
}
```

۴.۶ استراتژی‌های توزیع بار

Nginx از load balancing پشتیبانی می‌کند که درخواست‌ها را بین چندین سرور پشت‌صحنه توزیع می‌کند تا مقیاس‌پذیری و قابلیت اطمینان افزایش یابد.

۱.۴.۶ توزیع بار به روش Round Robin

به طور پیش‌فرض، Nginx از الگوریتم round-robin برای توزیع درخواست‌ها به طور یکنواخت بین سرورها استفاده می‌کند.

```
upstream backend {
    server 192.168.1.10;
    server 192.168.1.11;
    server 192.168.1.12;
}

server {
    listen 80;
    server_name example.com;

    location / {
        proxy_pass http://backend;
    }
}
```

۲.۴.۶ توزیع بار بر اساس کمترین اتصالات

درخواست‌ها را به سروری که کمترین تعداد اتصال فعال را دارد ارسال می‌کند.

```
upstream backend {
    least_conn;
    server 192.168.1.10;
    server 192.168.1.11;
    server 192.168.1.12;
}
```

۳.۴.۶ توزیع بار به روش IP Hash

این روش تضمین می‌کند که درخواست‌ها از همان IP کاربر همیشه به یک سرور خاص ارسال شود که برای session persistence مفید است.

```
upstream backend {
    ip_hash;
    server 192.168.1.10;
    server 192.168.1.11;
}
```

۴.۴.۶ بررسی وضعیت و انتقال به سرور سالم

اگر یکی از سرورها از کار بیافتد، Nginx به طور خودکار ترافیک را به سرورهای سالم هدایت می‌کند.

```
upstream backend {
    server 192.168.1.10 max_fails=3 fail_timeout=30s;
    server 192.168.1.11;
}
```

منابع

- https://nginx.org/en/docs/ - Nginx Documentation
- https://httpd.apache.org/docs/ - Apache HTTP Server Documentation
- https://home.cern/science/computing/birth-web - CERN: The Birth of the Web
- https://www.digitalocean.com/community/tutorials - DigitalOcean Tutorials (for practical guides on Nginx and web servers)
- Kurosu, H. (2018). *High-Performance Web Servers: Nginx and Apache*. Springer Publishing. (Book reference for understanding the detailed architecture and configurations of Nginx and Apache)
- Nginx Wiki. *Nginx Configuration Basics and Installation Guides*. Retrieved from `https://nginx.org/en/docs/`
- DigitalOcean Community Tutorials. *How to Install Nginx on Linux and Configure Server Blocks*. Retrieved from `https://www.digitalocean.com/community/tutorials`