Proxies

سرورهای پروکسی وب، در واقع واسطه هستند. پروکسی‌ها بین کلاینت‌ها و سرورها قرار گرفته و به‌عنوان «واسطه‌ها» عمل می‌کنند و پیام‌های HTTP را بین طرفین به عقب و جلو می‌برند. این فصل درباره سرورهای پراکسی HTTP، پشتیبانی ویژه از ویژگی‌های پروکسی و برخی از رفتارهای پیچیده‌ای که در هنگام استفاده از سرورهای پروکسی مشاهده خواهید کرد صحبت می‌کند.

مباحثی که در این فصل به آن‌ها پرداخته می‌شود عبارتند از:

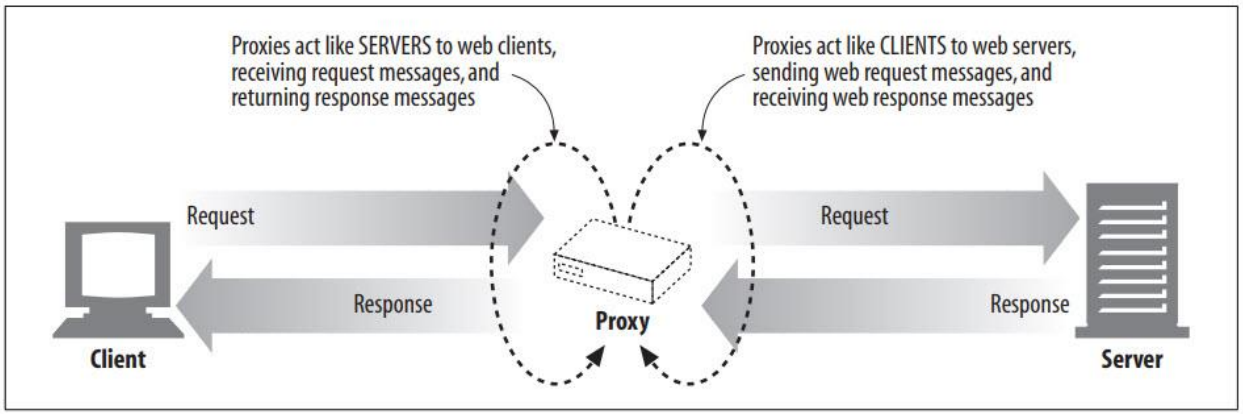
- توضیح پروکسی‌های HTTP ، نحوه قرار گرفتن آن‌ها در مقابل دروازه‌های و نحوه استقرار پروکسی‌ها
- توضیح برخی از راه‌هایی که پروکسی‌ها در آن مفید بوده و به ما کمک می‌کنند.
- توضیح نحوه استقرار پروکسی‌ها در شبکه‌های واقعی و نحوه هدایت ترافیک به سرورهای پروکسی
- نحوه پیکربندی مرورگر برای استفاده از پروکسی
- نشان دادن درخواست‌های پروکسی HTTP، تفاوت آن‌ها با درخواست‌های سرور و اینکه چگونه پروکسی‌ها می‌توانند رفتار مرورگرها را تغییر دهند.
- توضیح اینکه چگونه می‌توانید مسیر پیام‌های خود را از طریق زنجیره های سرورهای پروکسی با استفاده از هدرهای Via و متد TRACE ضبط کنید.
- توضیح کنترل دسترسی HTTP مبتنی بر پروکسی
- توضیح اینکه چگونه پروکسی‌ها می‌توانند بین کلاینت‌ها و سرورهایی که هر کدام از ویژگی‌ها و نسخه‌های متفاوتی پشتیبانی می‌کنند، تعامل داشته باشند.

Web Intermediaries

سرورهای پروکسی وب، واسطه هایی هستند که تراکنش‌ها را از طرف کلاینت انجام می‌دهند. بدون پروکسی وب، کلاینت‌های HTTP مستقیماً با سرورهای HTTP صحبت می‌کنند. با یک پروکسی وب، کلاینت در عوض با پروکسی صحبت می‌کند که خود از طرف کلاینت با سرور ارتباط برقرار می‌نماید. کلاینت همچنان تراکنش را کامل می‌کند، اما از طریق خدمات خوب سرور پروکسی.

سرورهای پروکسی HTTP، هم سرورهای وب و هم سرویس گیرندگان وب هستند. از آنجایی که سرویس گیرندگان HTTP پیام‌های درخواستی را به پروکسی‌ها ارسال می‌کنند، سرور پروکسی باید به درستی درخواست‌ها و اتصالات را مدیریت کند و پاسخ‌ها را درست مانند وب سرور بازگرداند. در همان زمان، خود پروکسی درخواست‌هایی را به سرورها ارسال می‌کند، بنابراین باید مانند یک کلاینت HTTP صحیح رفتار کند و

درخواست‌ها را ارسال نموده و پاسخ‌ها را دریافت کند (شکل زیر). اگر در حال ایجاد پروکسی HTTP خود هستید، باید قوانین را برای سرویس گیرندگان HTTP و سرورهای HTTP به دقت دنبال کنید.



Private and Shared Proxies

یک سرور پروکسی می‌تواند به یک کلاینت اختصاص داده شود یا بین بسیاری از کلاینت‌ها به اشتراک گذاشته شود. پروکسی‌هایی که به یک کلاینت اختصاص داده شده‌اند، پراکسی‌های خصوصی نامیده می‌شوند. پروکسی‌هایی که در بین کلاینت‌های متعدد به اشتراک گذاشته می‌شوند، پروکسی‌های عمومی نامیده می‌شوند.

Public proxies

اکثر پروکسی‌ها، پراکسی‌های عمومی و مشترک هستند. مدیریت یک پروکسی متمرکز مقرون به صرفه‌تر و آسان‌تر است. برخی از برنامه‌های پروکسی، مانند ذخیره‌سازی سرورهای پراکسی، زمانی که کاربران بیشتری به همان سرور پراکسی هدایت می‌شوند، مفیدتر می‌شوند، زیرا می‌توانند از درخواست‌های رایج بین کاربران استفاده کنند.

Private proxies

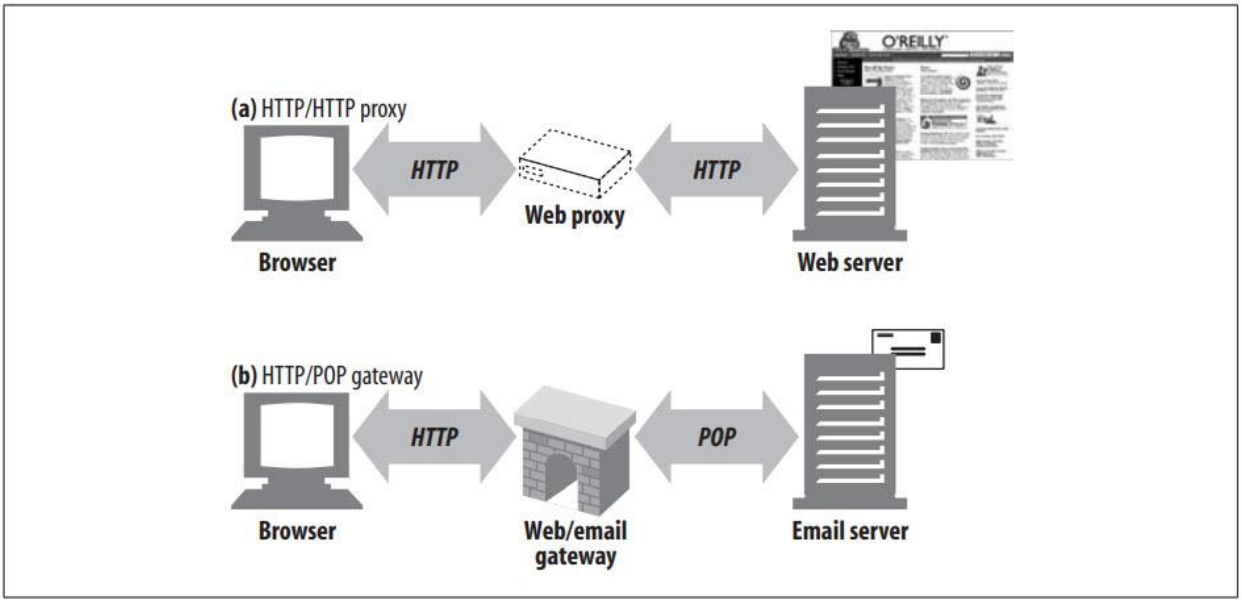
پروکسی‌های خصوصی اختصاصی چندان رایج نیستند، اما جایگاه خود را دارند، به خصوص زمانی که مستقیماً روی رایانه کلاینت اجرا شوند. برخی از محصولات دستیار مرورگر یا Browser Assistant و همچنین برخی از خدمات ISP، پروکسی‌های کوچکی را مستقیماً بر روی رایانه شخصی کاربر اجرا می‌کنند تا ویژگی‌های مرورگر را گسترش دهند، عملکرد را بهبود بخشند یا تبلیغات را برای خدمات رایگان ISP میزبانی کنند.

Proxies Versus Gateways

به بیان دقیق‌تر، پروکسی دو یا چند برنامه که با پروتکل یکسانی با یکدیگر صحبت می‌کنند را به هم متصل می‌کند، در حالی که Gateway دو یا چند طرف که پروتکل‌های متفاوتی دارند را به هم متصل

می‌کند. یک Gateway به عنوان یک "Protocol Converter" عمل می‌کند و به کلاینت اجازه می‌دهد تا تراکنش را با یک سرور انجام دهد، حتی زمانی که کلاینت و سرور دارای پروتکل‌های متفاوتی هستند.

شکل زیر تفاوت بین پروکسی‌ها و Gateway ها را نشان می‌دهد:



- دستگاه واسطه در بخش a شکل، یک پروکسی HTTP است، زیرا پروکسی از طریق HTTP هم با کلاینت و هم سرور صحبت می‌کند.
- دستگاه واسطه در بخش b شکل یک Gateway یا دروازه HTTP/POP است، زیرا یک فرانت اند HTTP را به پشتیبان ایمیل POP متصل می‌کند. این Gateway تراکنش‌های وب را به تراکنش‌های مناسب POP تبدیل می‌کند تا به کاربر اجازه دهد ایمیل را از طریق HTTP بخواند. برنامه‌های ایمیل مبتنی بر وب مانند Yahoo! Mail و MSN Hotmail دروازه‌های ایمیل HTTP یا HTTP Email Gateway هستند.

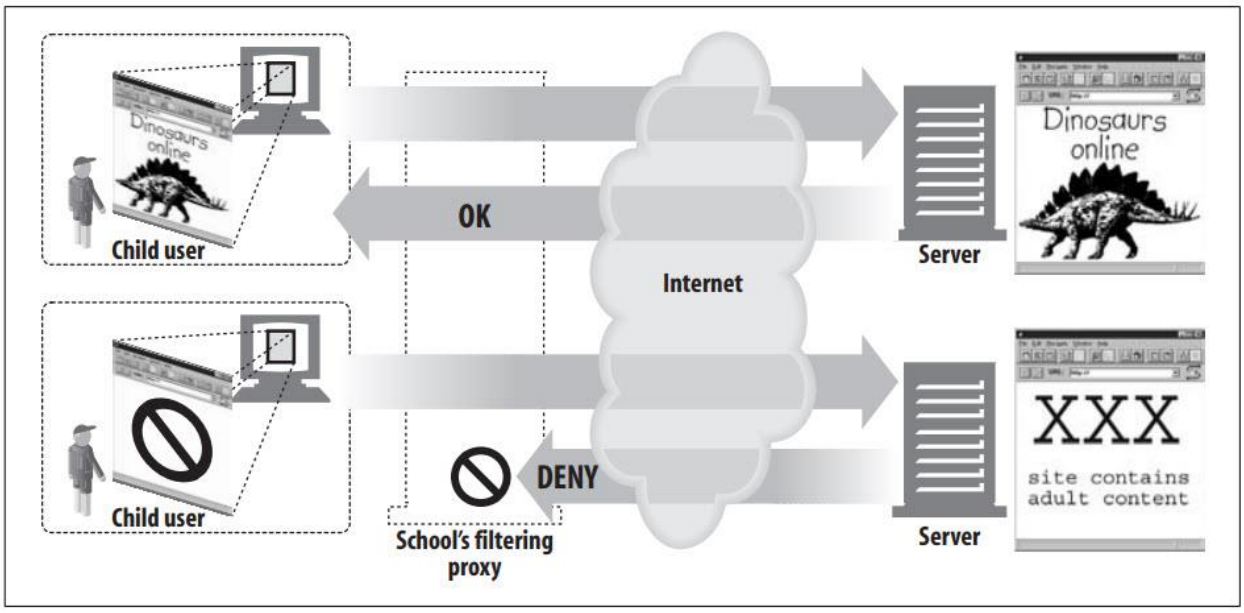
در عمل، تفاوت بین پروکسی‌ها و Gateway ها مبهم است. از آنجایی که مرورگرها و سرورها نسخه‌های مختلف HTTP را پیاده سازی می‌کنند، پروکسی‌ها اغلب مقداری از تبدیل پروتکل را انجام می‌دهند و سرورهای پروکسی تجاری عملکرد Gateway را برای پشتیبانی از پروتکل‌های امنیتی SSL، فایروال‌های SOCKS، دسترسی FTP و برنامه‌های کاربردی مبتنی بر وب پیاده سازی می‌کنند. در فصل ۸ بیشتر در مورد Gateway ها صحبت خواهیم کرد.

Why Use Proxies?

سرورهای پروکسی می‌توانند انواع کارهای مفید و کاربردی را انجام دهند. آن‌ها می‌توانند امنیت را بهبود بخشند، عملکرد را افزایش دهند و در هزینه صرفه جویی کنند. از آنجایی که سرورهای پروکسی می‌توانند تمام ترافیک عبوری HTTP را ببینند و لمس کنند، پروکسی‌ها می‌توانند ترافیک را برای پیاده‌سازی بسیاری از سرویس‌های وب با ارزش افزوده مفید، نظارت و تغییر دهند. در اینجا چند مورد از روش‌هایی که می‌توان از پروکسی‌ها استفاده نمود آورده شده است:

Child filter

مدارس ابتدایی از پراکسی‌های فیلتر برای مسدود کردن دسترسی به محتوای بزرگسالان استفاده می‌کنند، در حالی که دسترسی بدون مانع به سایت‌های آموزشی را فراهم می‌کنند. همانطور که در شکل زیر نشان داده شده است، پروکسی ممکن است اجازه دسترسی نامحدود به محتوای آموزشی را بدهد، اما از دسترسی به سایت‌هایی که برای کودکان نامناسب است جلوگیری کند.



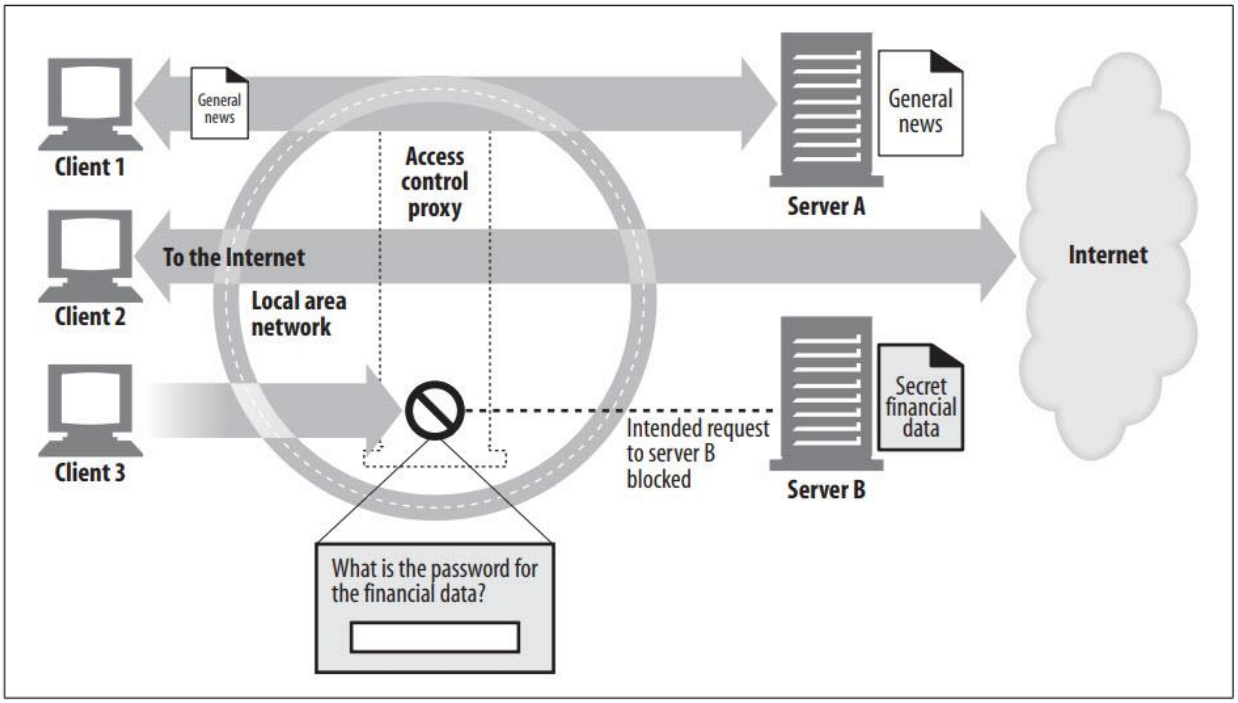
Document access controller

از سرورهای پروکسی می‌توان برای پیاده سازی یک استراتژی کنترل دسترسی یکنواخت در مجموعه بزرگی از وب سرورها و منابع وب و ایجاد یک مسیر حسابرسی استفاده کرد. این در تنظیمات شرکت‌های بزرگ یا سایر بوروکراسی‌های توزیع شده مفید است.

تمام کنترل‌های دسترسی را می‌توان بدون نیاز به بروزرسانی کنترل‌های دسترسی به طور مکرر در سرورهای وب متعدد، با مدل‌ها و ساختارهای مختلف، که توسط سازمان‌های مختلف مدیریت می‌شوند، بر روی سرور پروکسی متمرکز پیکربندی کرد.

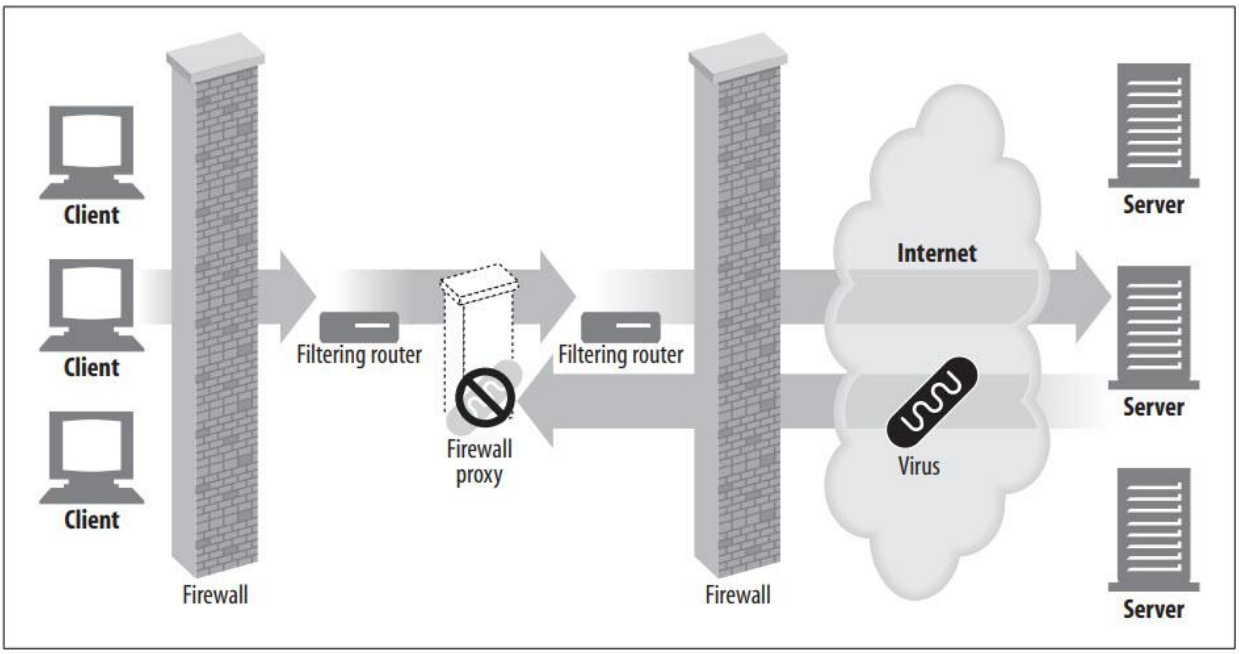
در شکل زیر، پروکسی کنترل دسترسی متمرکز:

- به کلاینت یک اجازه می‌دهد بدون محدودیت به صفحات اخبار از سرور A دسترسی داشته باشد.
- دسترسی نامحدود به محتوای اینترنت را برای کلاینت دو فراهم می‌کند.
- همچنین قبل از اجازه دسترسی به سرور B، به یک رمز عبور از کلاینت سه نیاز دارد.



Security firewall

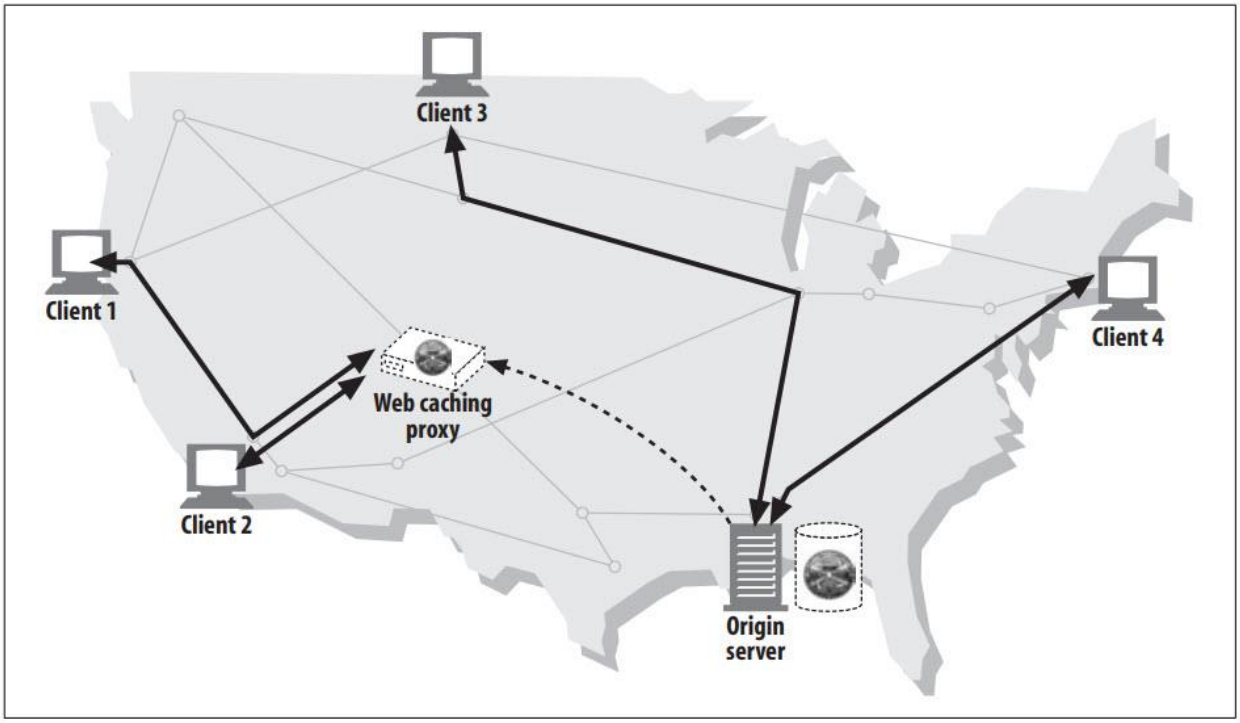
مهندسان امنیت شبکه اغلب از سرورهای پروکسی برای افزایش امنیت و اعمال محدودیت استفاده می‌کنند. بدین صورت که کدام پروتکل‌های سطح Application به داخل و خارج از یک سازمان، در یک نقطه امن در شبکه جریان پیدا کنند. آن‌ها همچنین می‌توانند قلاب‌هایی (hooks) برای بررسی دقیق آن ترافیک فراهم کنند، به همان شکلی که توسط پروکسی‌های وب و ایمیل حذف کننده ویروس استفاده می‌شود.



Web cache

Proxy cache، کپی‌های محلی اسناد محبوب را نگهداری می‌کند و آن‌ها را در صورت تقاضا ارائه می‌دهد و از ارتباطات اینترنتی کند و پرهزینه می‌کاهند.

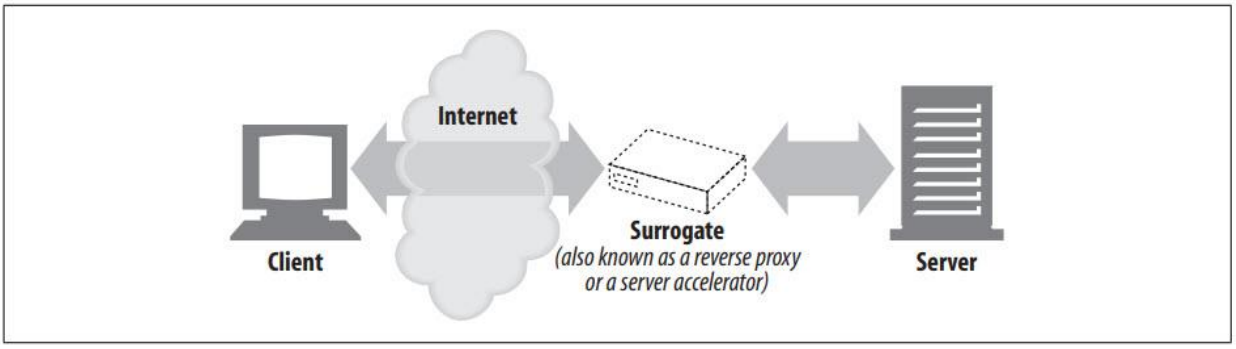
در شکل زیر، کلاینت‌های ۱ و ۲ به شی A از cache وب نزدیک دسترسی دارند، در حالی که کلاینت‌های ۳ و ۴ از سرور مبدا به سند دسترسی دارند.



Surrogate

پروکسی‌ها می‌توانند به عنوان سرورهای وب ظاهر شوند. این به‌اصطلاح جانشین‌ها (Surrogates) یا Reverse Proxy درخواست‌های وب سرور واقعی را دریافت می‌کنند، اما، برخلاف سرورهای وب، ممکن است ارتباط با سرورهای دیگر را برای یافتن محتوای درخواستی در صورت تقاضا آغاز نماید.

Surrogate ها ممکن است برای بهبود عملکرد وب سرورهای کند برای محتوای رایج استفاده شوند. در این پیکربندی، Surrogate ها اغلب شتاب دهنده سرور یا Server Accelerator نامیده می‌شوند. Surrogate ها همچنین می‌توانند در ارتباط با عملکرد مسیریابی محتوا برای ایجاد شبکه‌های توزیع شده از محتوای تکراری بر اساس تقاضا استفاده شوند.

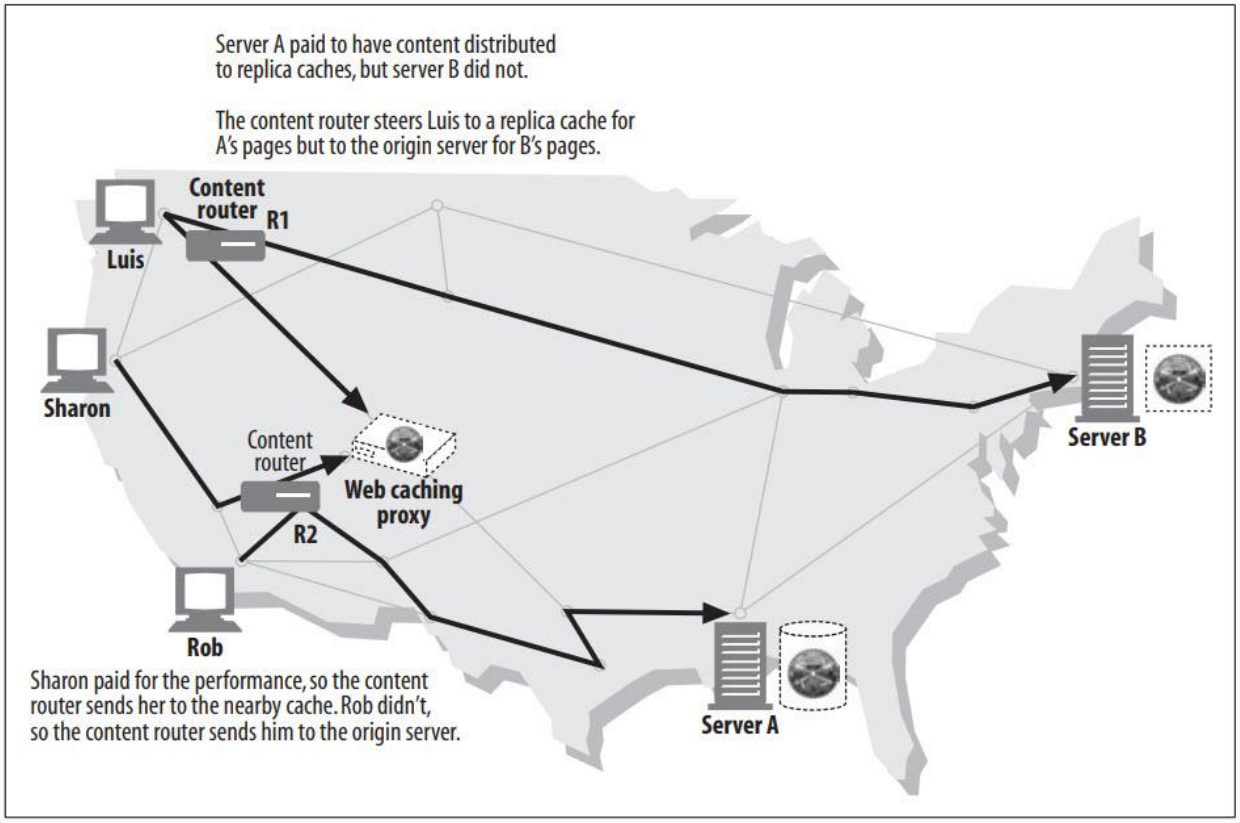


Content Router

سرورهای پروکسی می‌توانند به عنوان "content routers" عمل کنند و درخواست‌ها را بر اساس شرایط ترافیک اینترنت و نوع محتوا به سرورهای وب خاص منتقل کنند.

Content Router ها همچنین می‌توانند برای پیاده سازی پیشنهادات مختلف در سطح خدمات استفاده شوند.

برای مثال، اگر کاربر یا ارائه‌دهنده محتوا برای عملکرد بالاتر هزینه پرداخت کرده باشد، روترهای محتوا می‌توانند درخواست‌ها را به Cache مشابه ارسال کنند یا اگر کاربر برای یک سرویس فیلتر ثبت‌نام کرده باشد، درخواست‌های HTTP را از طریق پراکسی‌های فیلتر هدایت کنند. بسیاری از خدمات جالب را می‌توان با استفاده از پروکسی‌های مسیریابی محتوای تطبیقی ساخت.

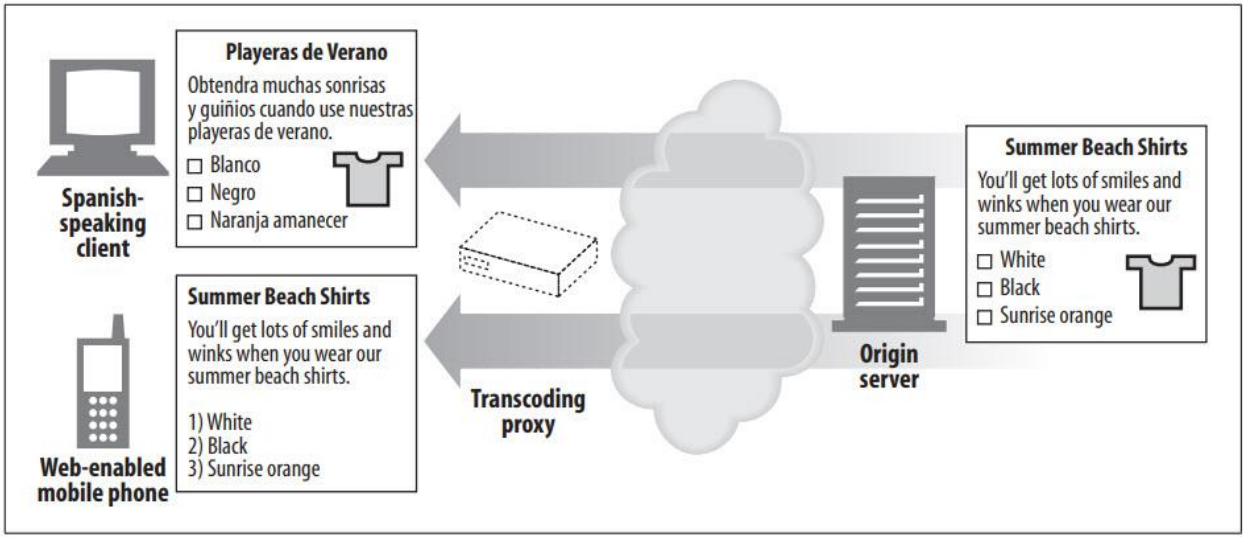


Transcoder

سرورهای پروکسی می‌توانند قالب متنی محتوا را قبل از تحویل آن به کلاینت‌ها تغییر دهند. این ترجمه شفاف (Transparent Translation) بین نمایش داده‌ها، Transcoding نامیده می‌شود.

پروکسی‌های Transcoding می‌توانند تصاویر GIF را در حین انتقال به تصاویر JPEG تبدیل کنند تا اندازه را کاهش دهند. همچنین می‌توان تصاویر را کوچک نموده و شدت رنگ را کاهش داد تا در تلویزیون قابل مشاهده باشند. به همین ترتیب، فایل‌های متنی را می‌توان فشرده کرد و خلاصه‌های متنی کوچکی از صفحات وب برای صفحات فعال اینترنت و تلفن‌های هوشمند تولید کرد. حتی برای پروکسی‌ها امکان تبدیل اسناد به زبان‌های خارجی در حین انتقال وجود دارد!

شکل زیر یک پروکسی Transcoding را نشان می‌دهد که متن انگلیسی را به متن اسپانیایی تبدیل می‌کند و همچنین صفحات HTML را به متن ساده‌تری که می‌تواند بر روی صفحه نمایش کوچک تلفن همراه نمایش داده شود، دوباره قالب‌بندی می‌کند.

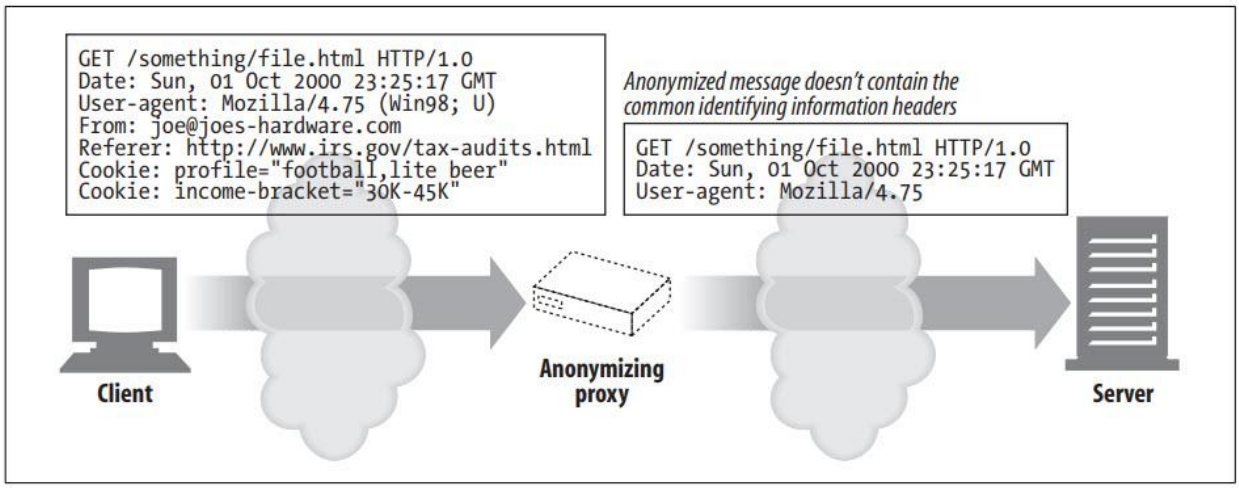


Anonymizer

پروکسی‌های ناشناس با حذف فعال مشخصه‌های شناسایی از پیام‌های HTTP (مانند آدرس IP کلاینت، از هدر، Referer، کوکی‌ها، شناسه‌های نشست URI) حریم خصوصی و ناشناس بودن را افزایش می‌دهند.

در شکل زیر، پروکسی ناشناس، تغییرات زیر را در پیام‌های کاربر برای افزایش حریم خصوصی ایجاد می‌کند:

- حذف نام کاربری سیستم و سیستم عامل کاربر از هدر User-Agent
- حذف هدر From برای محافظت از آدرس ایمیل کاربر
- حذف هدر Referer به منظور اینکه سایت‌های دیگری که کاربر بازدید کرده است مبهم باشد.
- حذف هدرهای کوکی برای حذف مشخصات و داده‌های هویتی



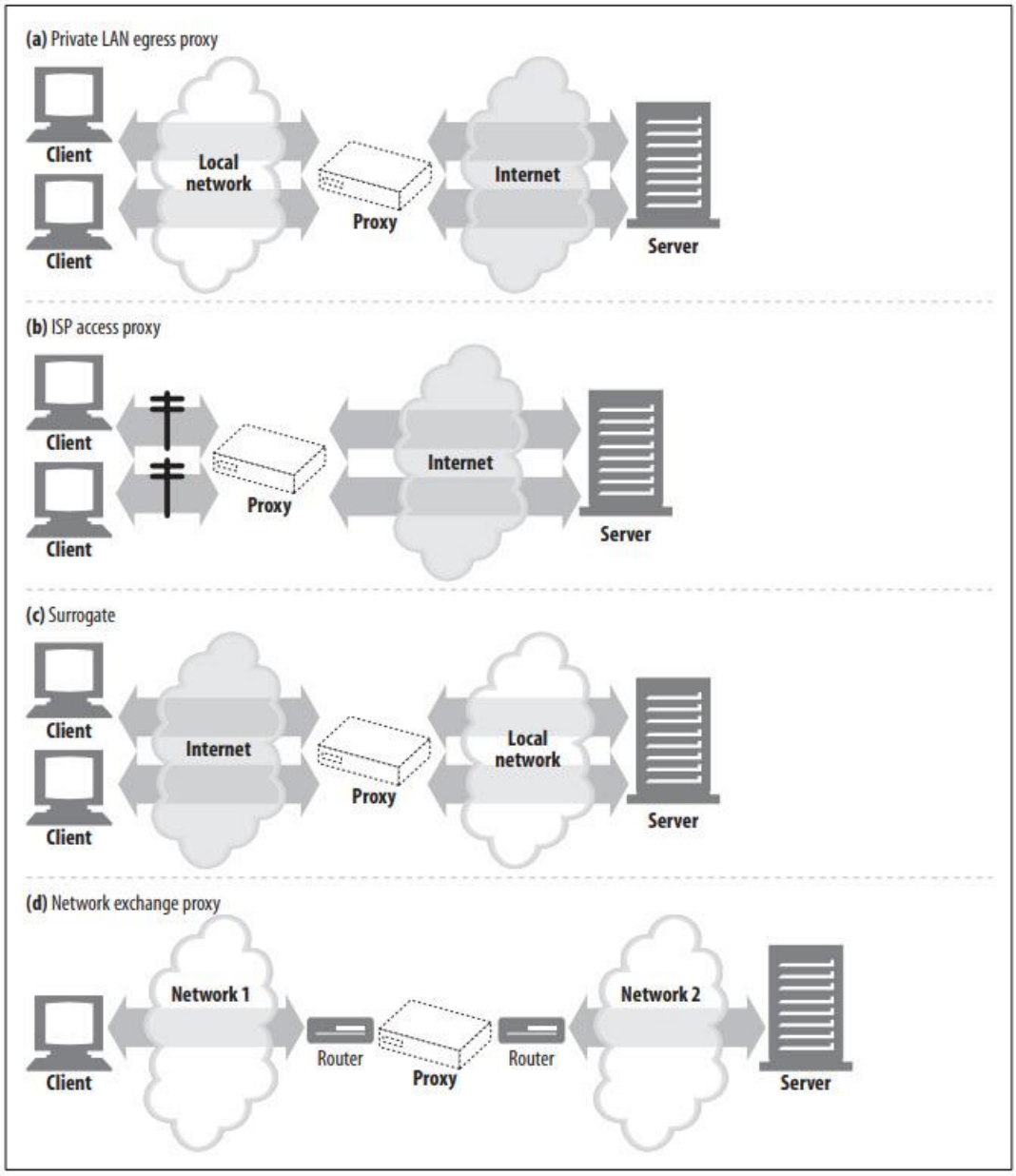
Where Do Proxies Go?

در بخش قبل توضیح داده شد که پروکسی‌ها چه کاری انجام می‌دهند. حالا بیایید در مورد جایی که پروکسی‌ها در معماری شبکه مستقر می‌شوند صحبت کنیم. در این بخش به موارد زیر می‌پردازیم:

- چگونه می‌توان پروکسی‌ها را در شبکه‌ها مستقر کرد.
- چگونه پروکسی‌ها می‌توانند با هم به صورت سلسله مراتبی Chain شوند.
- چگونه ترافیک در وهله اول به یک سرور پروکسی هدایت می‌شود.

Proxy Server Deployment

می‌توانید پروکسی‌ها را در انواع مکان‌ها، بسته به کاربرد مورد نظرشان، قرار دهید.



تصویر a بخش – Egress proxy

برای کنترل جریان ترافیک بین شبکه محلی و اینترنت، می‌توانید پروکسی‌ها را در نقاط خروجی شبکه‌های محلی قرار دهید. شما ممکن است از Egress proxy ها در یک شرکت به عنوان یک لایه محافظتی مشابه فایروال در برابر هکرهای مخرب خارج از شرکت یا کاهش هزینه‌های پهنای باند و بهبود عملکرد ترافیک اینترنت استفاده کنید. یک مدرسه ابتدایی ممکن است از یک Egress proxy فیلتر کننده برای جلوگیری از مرور محتوای نامناسب دانش آموزان استفاده کند.

Access (ingress) proxy – بخش b تصویر

پروکسی‌ها اغلب در نقاط دسترسی ISP قرار می‌گیرند و درخواست‌های کلی کلاینت‌ها را پردازش می‌کنند. ISP ها از پروکسی‌های Cache برای ذخیره یک کپی از اسناد محبوب، برای بهبود سرعت دانلود برای کاربران خود (به ویژه آن‌هایی که اتصالات پرسرعت دارند) و کاهش هزینه‌های پهنای باند اینترنت استفاده می‌کنند.

Surrogates – بخش c تصویر

پروکسی‌ها اغلب به عنوان Surrogate (که معمولاً به آن‌ها Reverse Proxy نیز گفته می‌شود) در لبه شبکه، در مقابل سرورهای وب مستقر می‌شوند، جایی که می‌توانند تمام درخواست‌های ارسال شده به وب سرور را وارد کنند و فقط در صورت لزوم از سرور وب درخواست منابع کنند. جایگزین‌ها می‌توانند ویژگی‌های امنیتی را به سرورهای وب اضافه کنند یا با قرار دادن حافظه پنهان وب سرور سریع در مقابل سرورهای وب کندتر، عملکرد را بهبود بخشند. Surrogate ها معمولاً نام و آدرس IP سرور وب را مستقیماً فرض می‌کنند، بنابراین همه درخواست‌ها به جای سرور به پروکسی می‌روند.

Network exchange proxy – بخش d تصویر

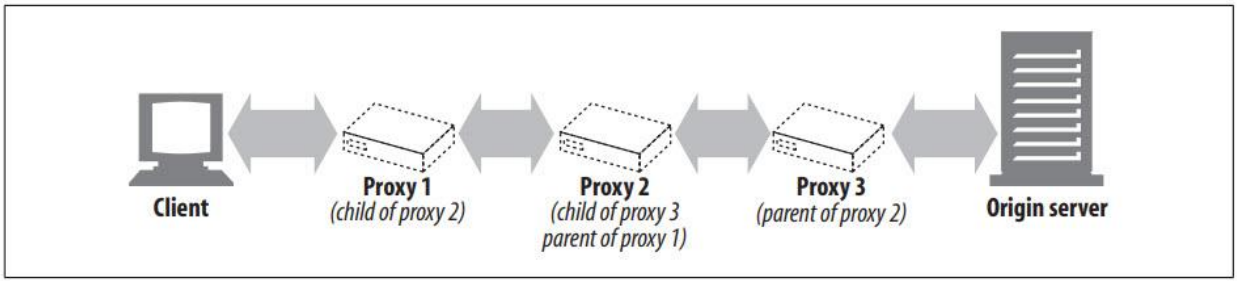
پروکسی‌ها را می‌توان در نقاط تبادل همتای اینترنت بین شبکه‌ها قرار داد تا ازدحام در اتصالات اینترنت از طریق Cache بکاهد و بر جریان ترافیک نظارت شود.

پروکسی‌های اصلی (Core Proxies) اغلب در جاهایی مستقر می‌شوند که پهنای باند اینترنت بسیار گران است (به ویژه در اروپا). برخی کشورها (مانند بریتانیا) نیز در حال ارزیابی استقرار پروکسی بحث برانگیز برای نظارت بر ترافیک اینترنت برای نگرانی‌های امنیت ملی هستند.

Proxy Hierarchies

پروکسی‌ها را می‌توان در زنجیره‌هایی به نام Proxy Hierarchies قرار داد. در یک Proxy Hierarchies، پیام‌ها از پروکسی به پروکسی دیگر منتقل می‌شوند تا زمانی که در نهایت به سرور مبدا برسند (و سپس از طریق پروکسی‌ها به کلاینت بازگردانده می‌شوند).

سرورهای پروکسی در یک Proxy Hierarchies دارای روابط والد و فرزندی هستند. پروکسی ورودی بعدی (نزدیکتر به سرور) والد و پروکسی خروجی بعدی (نزدیکتر به کلاینت) فرزند نامیده می‌شود. در شکل زیر، پروکسی ۱، پروکسی فرزند پروکسی ۲ است. به همین ترتیب، پروکسی ۲، پروکسی فرزند پروکسی ۳، و پروکسی ۳، پروکسی والد پروکسی ۲ است.



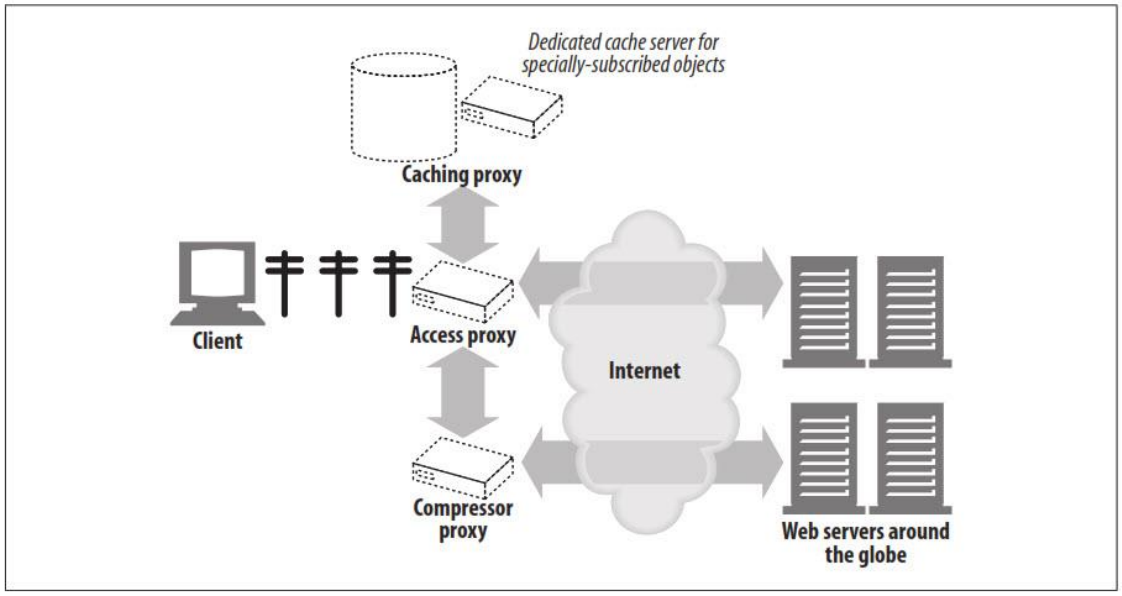
Proxy hierarchy content routing

سلسله مراتب پروکسی در شکل بالا ثابت است—پروکسی ۱ همیشه پیام‌ها را به پروکسی ۲ و پروکسی ۲ همیشه پیام‌ها را به پروکسی ۳ ارسال می‌کند. یک سرور پروکسی بر اساس عوامل بسیاری می‌تواند پیام‌ها را به مجموعه متنوع و متغیری از سرورهای پروکسی و سرورهای مبدا ارسال کند.

به عنوان مثال، در شکل زیر، مسیرهای Access Proxy به پروکسی‌های والد یا سرورهای مبدا در شرایط مختلف:

اگر شی درخواستی متعلق به سرور وب است که برای توزیع محتوا پول پرداخت کرده است، پروکسی می‌تواند درخواست را به یک سرور Cache مجاور هدایت کند که یا شی ذخیره‌شده را برمی‌گرداند یا اگر در دسترس نبود، آن را Fetch می‌کند.

اگر درخواست برای نوع خاصی از تصویر بود، Access Proxy ممکن است درخواست را به یک Compression Proxy اختصاصی هدایت کند که تصویر را Fetch می‌کند و سپس آن را فشرده می‌کند، بنابراین در یک مودم کند، سریع‌تر برای کلاینت دانلود می‌شود.



در اینجا چند نمونه دیگر از انتخاب والد پویا آورده شده است:

Load balancing

یک پروکسی فرزند ممکن است یک پروکسی والد را بر اساس میزان فعلی بار کاری والدین انتخاب کند تا بار را به اطراف پراکنده نماید.

Geographic proximity routing

یک پروکسی فرزند ممکن است والدینی را که مسئول منطقه جغرافیایی سرور مبدا است انتخاب کند.

Protocol/type routing

یک پروکسی فرزند ممکن است مسیریابی را بر اساس URI به والدین و سرورهای مبدأ مختلف انجام دهد. انواع خاصی از URI ها ممکن است باعث شوند که درخواست‌ها از طریق سرورهای پروکسی ویژه برای رسیدگی به پروتکل‌های خاص منتقل شوند.

Subscription-based routing

اگر ناشران برای خدمات با کارایی بالا پول بیشتری پرداخت کرده باشند، URI های آن‌ها ممکن است برای بهبود عملکرد به Cache یا موتورهای فشرده سازی بزرگ هدایت شوند.

منطق مسیریابی والدین پویا در محصولات مختلف، از جمله فایل‌های پیکربندی، زبان‌های برنامه‌نویسی و افزونه‌های اجرایی پویا، به‌طور متفاوتی پیاده‌سازی می‌شود.

How Proxies Get Traffic

از آنجایی که کلاینت‌ها معمولاً مستقیماً با سرورهای وب صحبت می‌کنند، باید توضیح دهیم که چگونه ترافیک HTTP در وهله اول به یک پروکسی راه پیدا می‌کند. چهار راه متداول برای ایجاد ترافیک کلاینت به یک پروکسی وجود دارد:

Modify the client

بسیاری از کلاینت‌های وب، از جمله مرورگرهای Netscape و Microsoft، از پیکربندی دستی و خودکار پروکسی پشتیبانی می‌کنند. اگر یک کلاینت برای استفاده از یک سرور پروکسی پیکربندی شده باشد، کلاینت درخواست‌های HTTP را مستقیماً و عمداً به جای سرور مبدا به پروکسی ارسال می‌کند (بخش a شکل).

Modify the network

چندین تکنیک وجود دارد که در آن زیرساخت شبکه بدون اطلاع یا مشارکت کلاینت، ترافیک وب را به یک پروکسی هدایت می‌کند. این رهگیری (Interception) معمولاً به دستگاه‌های سوئیچینگ و مسیریابی متکی است که ترافیک HTTP را تماشا می‌کنند، آن را رهگیری می‌کنند و بدون اطلاع کلاینت، ترافیک را به یک پروکسی منتقل می‌کنند (بخش b شکل). این موضوع را Intercepting Proxy می‌گویند. پروکسی‌های رهگیری معمولاً «Transparent Proxies» نامیده می‌شوند، زیرا شما بدون اطلاع از حضورشان، به آن‌ها متصل می‌شوید. از آنجایی که اصطلاح «Transparency» قبلاً در مشخصات HTTP برای نشان دادن عملکردهایی که رفتار معنایی را تغییر نمی‌دهند استفاده می‌شود، جامعه استاندارد پیشنهاد می‌کند از عبارت «Interception» برای ضبط ترافیک استفاده شود.

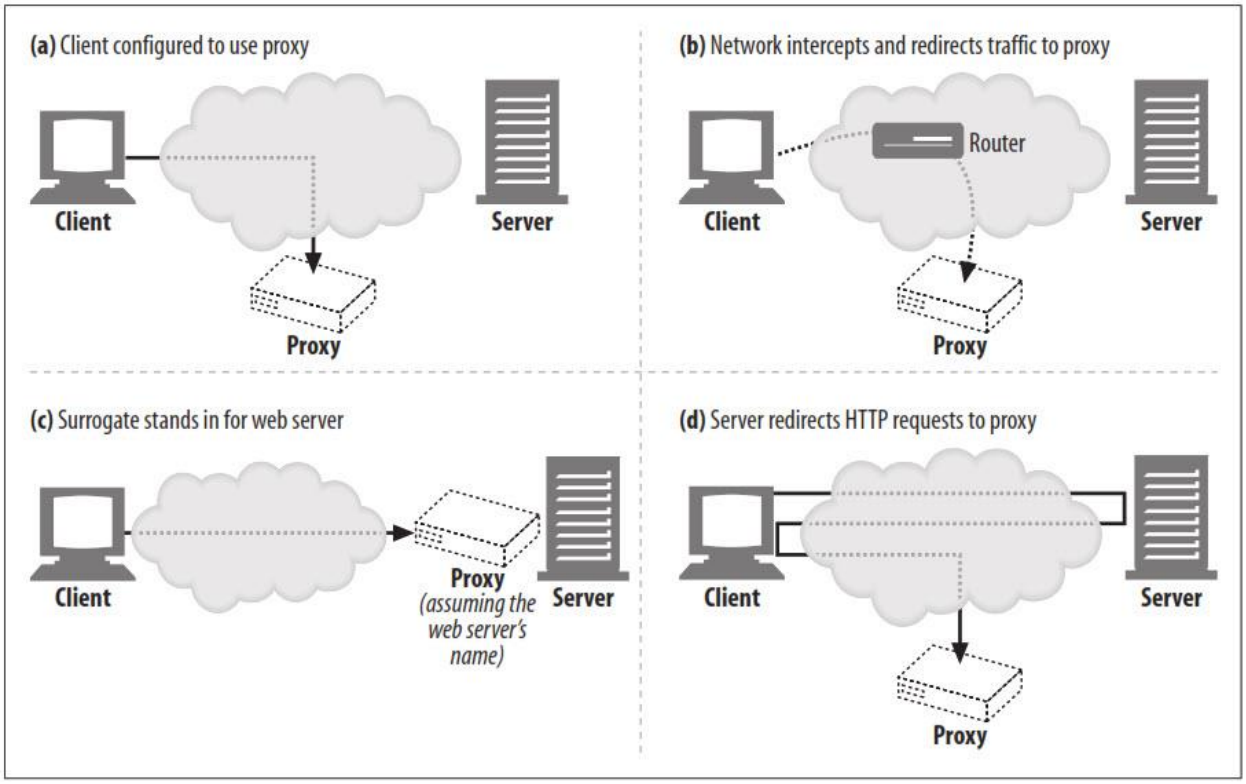
Modify the DNS namespace

Surrogates، که سرورهای پروکسی هستند که در مقابل سرورهای وب قرار می‌گیرند، نام و آدرس IP سرور وب را مستقیماً فرض می‌کنند، بنابراین همه درخواست‌ها به جای سرور به آن‌ها می روند (بخش c شکل). این را می‌توان با ویرایش دستی جداول نامگذاری DNS یا با استفاده از سرورهای DNS پویا خاص که پروکسی یا سرور مناسب را برای استفاده بر حسب تقاضا محاسبه می‌کنند، ترتیب داد. در برخی از نصب‌ها، آدرس IP و نام سرور واقعی تغییر می‌کند و به Surrogate آدرس و نام قبلی داده می‌شود.

Modify the web server

برخی از وب سرورها نیز می‌توانند پیکربندی شوند تا درخواست‌های کلاینت را به یک پروکسی با ارسال یک فرمان تغییر مسیر HTTP (کد پاسخ ۳۰۵) به کلاینت هدایت کنند. پس از دریافت تغییر مسیر، کلاینت با پروکسی تراکنش می‌کند (بخش d شکل).

بخش بعدی نحوه پیکربندی کلاینت‌ها برای ارسال ترافیک به پروکسی‌ها را توضیح می‌دهد. در فصل ۲۰ نحوه پیکربندی شبکه، DNS و سرورها برای هدایت ترافیک به سرورهای پروکسی توضیح داده خواهد شد.



Client Proxy Settings

همه مرورگرهای وب مدرن به شما این امکان را می‌دهند تا پیکربندی لازم برای استفاده از پروکسی‌ها را انجام دهید. در واقع، بسیاری از مرورگرها راه‌های مختلفی را برای پیکربندی پروکسی‌ها ارائه می‌دهند، از جمله:

Manual configuration

شما به صراحت یک پروکسی برای استفاده تنظیم کرده اید.

Browser preconfiguration

فروشنده یا توزیع کننده مرورگر به صورت دستی تنظیمات پروکسی مرورگر (یا هر سرویس گیرنده وب دیگر) را قبل از تحویل آن به کلاینت، از قبل پیکربندی می‌کند.

Proxy auto-configuration (PAC)

شما یک URI به فایل پیکربندی خودکار پروکسی جاوا اسکریپت (PAC) ارائه می‌دهید. کلاینت فایل جاوا اسکریپت را Fetch می‌کند و آن را اجرا می‌نماید تا تصمیم بگیرد که آیا باید از پروکسی استفاده کند و اگر چنین است، از کدام سرور پروکسی استفاده شود.

WPAD proxy discovery

برخی از مرورگرها از پروتکل Web Proxy Autodiscovery Protocol (WPAD) پشتیبانی می‌کنند، که به طور خودکار یک "Configuration Server" را شناسایی می‌کند که مرورگر می‌تواند یک فایل پیکربندی خودکار را از آن دانلود کند.

Client Proxy Configuration: Manual

بسیاری از سرویس گیرندگان وب به شما اجازه می‌دهند تا پروکسی‌ها را به صورت دستی پیکربندی کنید. هر دو Netscape Navigator و Microsoft Internet Explorer پشتیبانی مناسبی برای پیکربندی پروکسی دارند.

Client Proxy Configuration: PAC Files

پیکربندی دستی پروکسی ساده اما غیر قابل انعطاف است. شما می‌توانید تنها یک سرور پروکسی را برای همه محتوا مشخص کنید و هیچ پشتیبانی از failover وجود ندارد. پیکربندی دستی پروکسی همچنین منجر به مشکلات مدیریتی برای سازمان‌های بزرگ می‌شود. با پایگاه بزرگی از مرورگرهای پیکربندی شده، پیکربندی مجدد هر مرورگر در صورت نیاز به ایجاد تغییرات دشوار یا غیرممکن است.

فایل‌های پیکربندی خودکار پروکسی (PAC) راه‌حلی پویاتر برای پیکربندی پروکسی هستند، زیرا آن‌ها برنامه‌های کوچک جاوا اسکریپت هستند که تنظیمات پروکسی را در لحظه محاسبه می‌کنند. هر بار که به یک سند دسترسی پیدا می‌شود، یک تابع جاوا اسکریپت، سرور پروکسی مناسب را انتخاب می‌کند.

برای استفاده از فایل‌های PAC، مرورگر خود را با URI فایل PAC جاوا اسکریپت پیکربندی کنید (پیکربندی شبیه به پیکربندی دستی است، اما شما یک URI را در کادر «پیکربندی خودکار» ارائه می‌کنید). مرورگر فایل PAC را از این URI دریافت می‌کند و از منطق جاوا اسکریپت برای محاسبه سرور پراکسی مناسب

برای هر دسترسی استفاده می‌کند. فایل‌های PAC معمولاً دارای پسوند pac. و نوع MIME «application/x-ns-proxy-autoconfig» هستند.

هر فایل PAC باید تابعی به نام FindProxyForURL(url,host) تعریف کند که سرور پروکسی مناسب را برای دسترسی به URI محاسبه می‌کند. مقدار برگشتی تابع می‌تواند هر یک از مقادیر جدول زیر باشد.

FindProxyForURL return value	Description
DIRECT	Connections should be made directly, without any proxies.
PROXY host:port	The specified proxy should be used.
SOCKS host:port	The specified SOCKS server should be used.

فایل PAC در مثال زیر یک پروکسی را برای تراکنش‌های HTTP، یک پروکسی دیگر را برای تراکنش‌های FTP و اتصالات مستقیم را برای همه انواع دیگر تراکنش‌ها الزامی می‌کند.

```
function FindProxyForURL(url, host) {
    if (url.substring(0,5) == "http:") {
        return "PROXY http-proxy.mydomain.com:8080";
    } else if (url.substring(0,4) =="ftp:") {
        return "PROXY ftp-proxy.mydomain.com:8080";
    } else {
        return "DIRECT";
    }
}
```

Client Proxy Configuration: WPAD

مکانیزم دیگر برای پیکربندی مرورگر، پروتکل کشف خودکار پروکسی وب (WPAD) است. WPAD الگوریتمی است که از یک استراتژی افزایشی مکانیسم‌های کشف برای یافتن خودکار فایل PAC مناسب برای مرورگر استفاده می‌کند. کلاینتی که پروتکل WPAD را پیاده سازی می‌کند:

- از WPAD برای پیدا کردن PAC URI استفاده می‌کند.
- فایل PAC را با توجه به URI واکشی می‌کند.
- فایل PAC را برای تعیین سرور پروکسی اجرا می‌کند.
- برای درخواست‌ها از سرور پروکسی استفاده می‌کند.

WPAD از یک سری تکنیک‌های کشف منبع برای تعیین فایل PAC مناسب استفاده می‌کند. تکنیک‌های کشف چندگانه در این بخش استفاده می‌شود، زیرا همه سازمان‌ها نمی‌توانند از همه تکنیک‌ها استفاده کنند. WPAD هر تکنیک را یک به یک امتحان می‌کند تا زمانی که موفق شود.

مشخصات WPAD فعلی تکنیک‌های زیر را به ترتیب تعریف می‌کند:

- Dynamic Host Discovery Protocol (DHCP)
- Service Location Protocol (SLP)
- DNS well-known hostnames
- DNS SRV records
- DNS service URIs in TXT records

Tricky Things About Proxy Requests

این بخش برخی از جنبه‌های پیچیده و سوء تفاهم شده درخواست‌های سرور پروکسی را توضیح می‌دهد، از جمله:

- تفاوت URI ها در درخواست‌های پروکسی با درخواست‌های سرور
- چگونه پروکسی های رهگیری(Intercepting) و معکوس(Reverse) می‌توانند اطلاعات میزبان سرور را مبهم کنند؟
- قوانین اصلاح URI
- چگونه پروکسی‌ها بر تکمیل خودکار URI هوشمندانه مرورگر یا ویژگی‌های گسترش نام میزبان تأثیر می‌گذارند؟

Proxy URIs Differ from Server URIs

وب سرور و پیام‌های پروکسی وب دارای Syntax یکسان هستند، با یک استثنا. URI در پیام درخواست HTTP زمانی متفاوت است که یک کلاینت به جای پروکسی، درخواست را به سرور ارسال کند.

هنگامی که یک سرویس گیرنده درخواستی را به یک وب سرور ارسال می‌کند، خط درخواست تنها حاوی یک URI جزئی (بدون طرح، میزبان یا پورت) است، همانطور که در مثال زیر نشان داده شده است:

```
GET /index.html HTTP/1.0
    User-Agent: SuperBrowserv1.3
```

با این حال، هنگامی که یک مشتری درخواستی را به یک پروکسی ارسال می‌کند، خط درخواست شامل URI کامل است. مثلا:

GET http://www.marys-antiques.com/index.html HTTP/1.0

User-Agent: SuperBrowser v1.3

چرا دو فرمت درخواست متفاوت داریم، یکی برای پروکسی‌ها و دیگری برای سرورها؟ در طراحی اصلی HTTP، کلاینت‌ها مستقیماً با یک سرور واحد صحبت می‌کردند. میزبانی مجازی وجود نداشت و هیچ پیش بینی ای برای پروکسی‌ها در نظر گرفته نشده بود. از آنجایی که یک سرور واحد نام میزبان و پورت خود را می‌داند، برای جلوگیری از ارسال اطلاعات اضافی، کلاینت‌ها فقط URI جزئی را بدون طرح و میزبان (و پورت) ارسال می‌کنند.

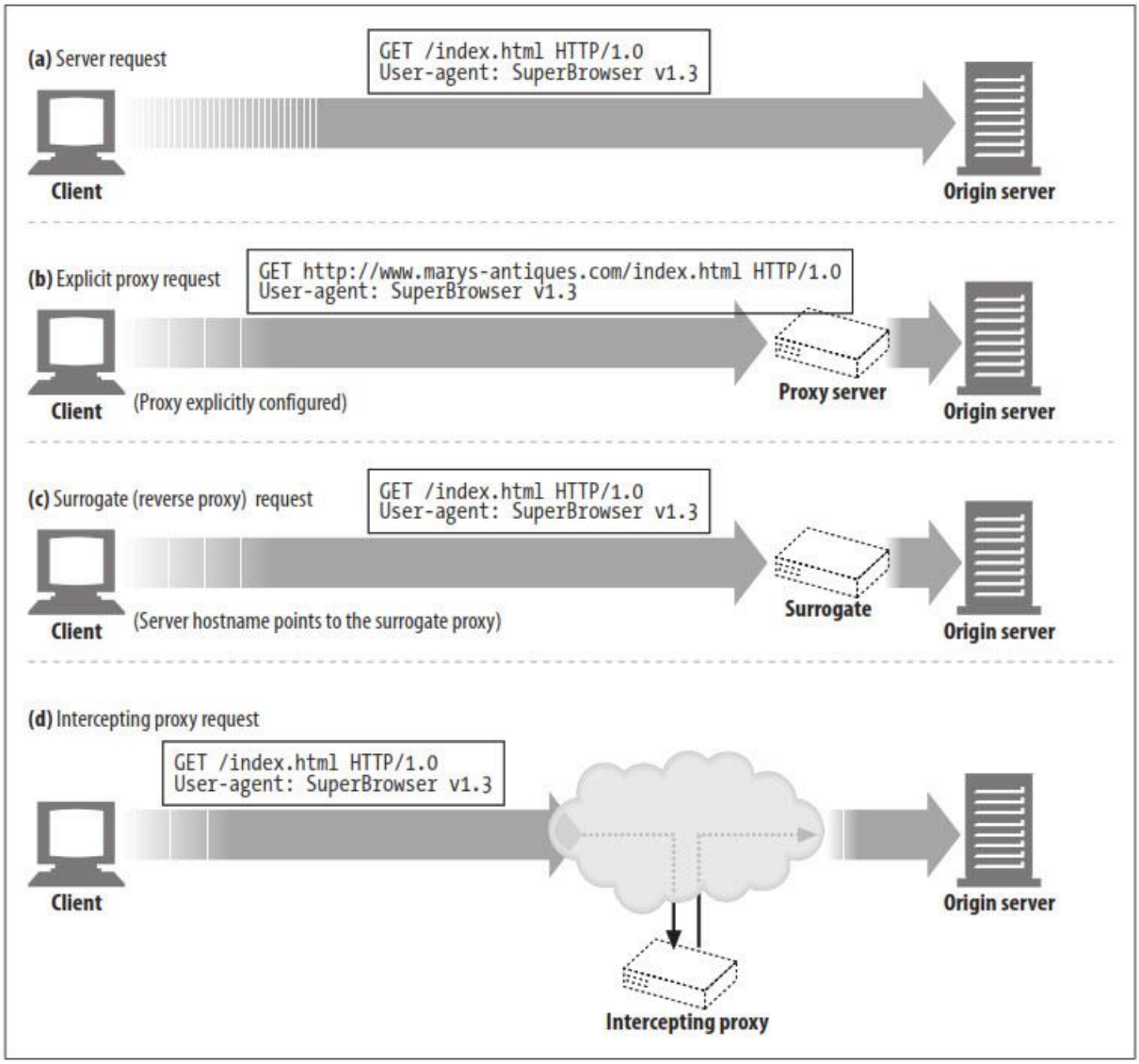
هنگامی که پروکسی‌ها ظاهر شدند، URIهای جزئی به یک مشکل تبدیل شدند. پروکسی‌ها باید نام سرور مقصد را بدانند تا بتوانند اتصالات خود را با سرور برقرار کنند و Gateway های مبتنی بر پروکسی برای اتصال به منابع FTP و سایر طرح‌ها به طرح URI نیاز داشتند.

HTTP/1.0 با نیاز به URI کامل برای درخواست‌های پروکسی، مشکل را حل کرد، اما URIهای جزئی را برای درخواست‌های سرور حفظ کرد (سرورهای زیادی از قبل مستقر شده بودند تا همه آن‌ها را تغییر دهند تا از URI های کامل پشتیبانی کنند).

HTTP/1.1 در حال حاضر به سرورها نیاز دارد که URIهای کامل را برای درخواست‌های پروکسی و سرور مدیریت کنند، اما در عمل، بسیاری از سرورهای مستقر هنوز فقط URI های جزئی را می‌پذیرند.

بنابراین باید URI های جزئی را به سرورها و URI های کامل را به پروکسی‌ها ارسال کنیم. در مورد تنظیمات پروکسی کلاینت به طور صریح پیکربندی شده، کلاینت می‌داند که چه نوع درخواستی را صادر کند:

- هنگامی که کلاینت برای استفاده از پروکسی تنظیم نشده است، URI جزئی را ارسال می‌کند (بخش a از شکل زیر).
- هنگامی که کلاینت برای استفاده از پروکسی تنظیم می‌شود، URI کامل را ارسال می‌کند (بخش b از شکل زیر).



The Same Problem with Virtual Hosting

مشکل پروکسی «از دست رفته طرح/میزبان/پورت» همان مشکلی است که سرورهای وب میزبان مجازی با آن مواجه هستند. وب سرورهای مجازی میزبان وب سرور فیزیکی یکسانی را در بین بسیاری از وب سایت‌ها به اشتراک می‌گذارند. وقتی درخواستی برای URI جزئی index.html وارد می‌شود، وب سرور مجازی باید نام میزبان وب‌سایت مورد نظر را بداند.

با وجود مشابه بودن مشکلات، آن‌ها به روش‌های مختلفی حل شدند:

- Explicit Proxy ها مشکل را با نیاز به یک URI کامل در پیام درخواست حل می‌کنند.

- وب سرورهای میزبان مجازی (Virtually Hosted Web Server) برای حمل اطلاعات میزبان و پورت به هدر Host نیاز دارند.

Intercepting Proxies Get Partial URIs

تا زمانی که کلاینت‌ها HTTP را به درستی پیاده‌سازی کنند، URIهای کامل را در درخواست‌ها به پروکسی‌هایی که به صورت Explicit پیکربندی شده‌اند ارسال می‌کنند. این بخشی از مشکل را حل می‌کند، اما یک نکته وجود دارد:

یک کلاینت همیشه نمی‌داند که با یک پروکسی صحبت می‌کند، زیرا ممکن است برخی از پروکسی‌ها برای کلاینت نامرئی باشند. حتی اگر کلاینت برای استفاده از پروکسی پیکربندی نشده باشد، ترافیک کلاینت همچنان ممکن است از طریق یک پروکسی Surrogate یا Intercepting انجام شود. در هر دوی این موارد، کلاینت فکر می‌کند که با یک وب سرور صحبت می‌کند و URI کامل را ارسال نمی‌کند:

Surrogate، همانطور که قبلا توضیح داده شد، یک سرور پروکسی است که معمولاً با فرض نام میزبان یا آدرس IP آن، جای سرور مبدا را می‌گیرد. این مدل پروکسی، درخواست وب سرور را دریافت می‌کند و ممکن است پاسخ‌های کش شده یا درخواست‌های پروکسی را به سرور واقعی ارائه دهد. یک کلاینت نمی‌تواند یک جانشین را از یک وب سرور تشخیص دهد، بنابراین URI های جزئی را ارسال می‌کند (بخش c شکل پیشین).

یک پروکسی Intercepting یک سرور پروکسی در جریان شبکه است که ترافیک را از کلاینت به سرور ربوده و یا یک پاسخ کش را ارائه می‌دهد یا آن را پروکسی می‌کند. از آنجایی که پروکسی Intercepting ترافیک کلاینت به سرور را ربوده، URI های جزئی را دریافت می‌کند که به سرورهای وب ارسال می‌شود (بخش d شکل پیشین).

Proxies Can Handle Both Proxy and Server Requests

به دلیل راه‌های مختلفی که می‌توان ترافیک را به سرورهای پروکسی هدایت کرد، سرورهای پروکسی همه منظوره باید از URIهای کامل و URIهای جزئی در پیام‌های درخواستی پشتیبانی کنند. اگر یک درخواست پروکسی Explicit است، پروکسی باید از URI کامل استفاده کند یا اگر درخواست سرور وب است از URI جزئی و هدر میزبان مجازی استفاده نماید.

قوانین استفاده از URI های کامل و جزئی عبارتند از:

- اگر یک URI کامل ارائه شده باشد، پروکسی باید از آن استفاده کند.

- اگر یک URI جزئی ارائه شده باشد و یک هدر میزبان وجود داشته باشد، باید از هدر میزبان برای تعیین نام سرور مبدا و شماره پورت استفاده شود.
- اگر یک URI جزئی ارائه شده باشد و هدر میزبان وجود نداشته باشد، سرور مبدا باید به روش دیگری تعیین شود:
  - اگر پروکسی Surrogate باشد و برای یک سرور مبدأ قرار گرفته باشد، پروکسی را می‌توان با آدرس سرور واقعی و شماره پورت پیکربندی کرد.
  - اگر ترافیک Intercepting شده باشد و Interceptor آدرس IP و پورت اصلی را در دسترس قرار دهد، پروکسی می‌تواند از آدرس IP و شماره پورت فناوری interception استفاده کند.
  - اگر همه چیز شکست بخورد، پروکسی اطلاعات کافی برای تعیین سرور مبدا ندارد و باید یک پیغام خطا برگرداند (اغلب به کاربر پیشنهاد می‌کند از یک مرورگر مدرن که از هدرهای میزبان پشتیبانی می‌کند استفاده نماید).

In-Flight URI Modification

سرورهای پروکسی باید در مورد تغییر URI درخواست هنگام ارسال پیام‌ها بسیار مراقب باشند. تغییرات جزئی در URI، حتی اگر خوش خیم به نظر برسند، ممکن است مشکلات قابلیت همکاری با سرورهای پایین دستی را ایجاد کنند.

به طور خاص، برخی از پروکسی‌ها که با نام canonicalize شناخته می‌شوند URIها را قبل از ارسال به پرش بعدی به شکلی استاندارد متعارف می‌کنند. تغییرات ظاهراً خوش‌خیم، مانند جایگزینی پورت‌های HTTP پیش‌فرض با یک «:۸۰» صریح، یا تصحیح URI ها با جایگزینی کاراکترهای رزرو شده غیرقانونی با جایگزین‌هایی که به درستی Escape شده‌اند، می‌توانند باعث ایجاد مشکلاتی در عملکرد شوند.

به طور کلی، سرورهای پروکسی باید تلاش کنند تا حد امکان Tolerant داشته باشند. آن‌ها نباید "پلیس پروتکل" باشند که به دنبال اجرای دقیق انطباق با پروتکل هستند، زیرا این امر می‌تواند شامل اختلال قابل توجهی در خدمات قبلی باشد.

به طور خاص، مشخصات HTTP، Intercepting Proxy های عمومی را از بازنویسی قسمت‌های مسیر مطلق URIها هنگام ارسال آن‌ها منع می‌کند. تنها استثنا این است که آن‌ها می‌توانند یک مسیر خالی را با "/" جایگزین کنند.

URI Client Auto-Expansion and Hostname Resolution

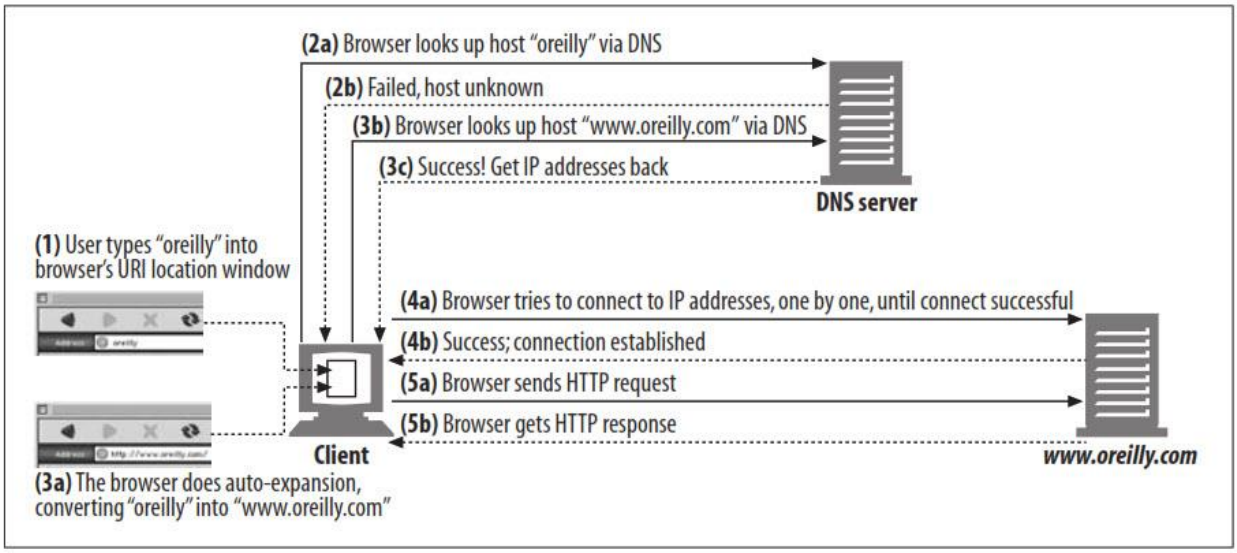
مرورگرها، بسته به وجود یا نبودن پروکسی، URI های درخواست را به طور متفاوتی Resolve می‌کنند. بدون پروکسی، مرورگر URI شما را می‌گیرد و سعی می‌کند آدرس IP مربوطه را پیدا کند. اگر نام میزبان پیدا شود، مرورگر آدرس‌های IP مربوطه را امتحان می‌کند تا زمانی که اتصال موفقیت‌آمیز انجام شود.

اما اگر میزبان پیدا نشد، بسیاری از مرورگرها سعی می‌کنند تا مقداری Expansion خودکار نام‌های میزبان را ارائه کنند، در صورتی که مخفف میزبان را تایپ کرده باشید (به Expandomatic URLs در فصل ۲ مراجعه کنید):

- بسیاری از مرورگرها سعی می‌کنند یک "www" اضافه کنند. پیشوند و پسوند «.com»، در صورتی که فقط قسمت میانی نام یک وب‌سایت رایج را وارد کرده باشید (به عنوان مثال، اجازه می‌دهد افراد به جای «www.yahoo.com»، عبارت yahoo را وارد کنند).
- برخی از مرورگرها حتی URI غیرقابل Resolve شما را به یک سایت شخص ثالث ارسال می‌کنند، که سعی می‌کند اشتباهات املایی را تصحیح کند و URI هایی را که ممکن است مد نظر شما باشد پیشنهاد می‌کند.
- علاوه بر این، پیکربندی DNS در اکثر سیستم‌ها به شما امکان می‌دهد فقط پیشوند نام میزبان را وارد کنید و DNS به طور خودکار، دامنه را جستجو می‌کند. اگر در دامنه "oreilly.com" هستید و نام میزبان "host7" را تایپ کنید، DNS به طور خودکار سعی می‌کند با "host7.oreilly.com" مطابقت داشته باشد. این یک نام میزبان کامل و معتبر نیست.

URI Resolution Without a Proxy

شکل زیر نمونه ای از گسترش خودکار نام میزبان مرورگر بدون پروکسی را نشان می‌دهد. در مراحل 2a تا 3c، مرورگر تغییرات نام میزبان را جستجو می‌کند تا زمانی که یک نام میزبان معتبر پیدا شود.



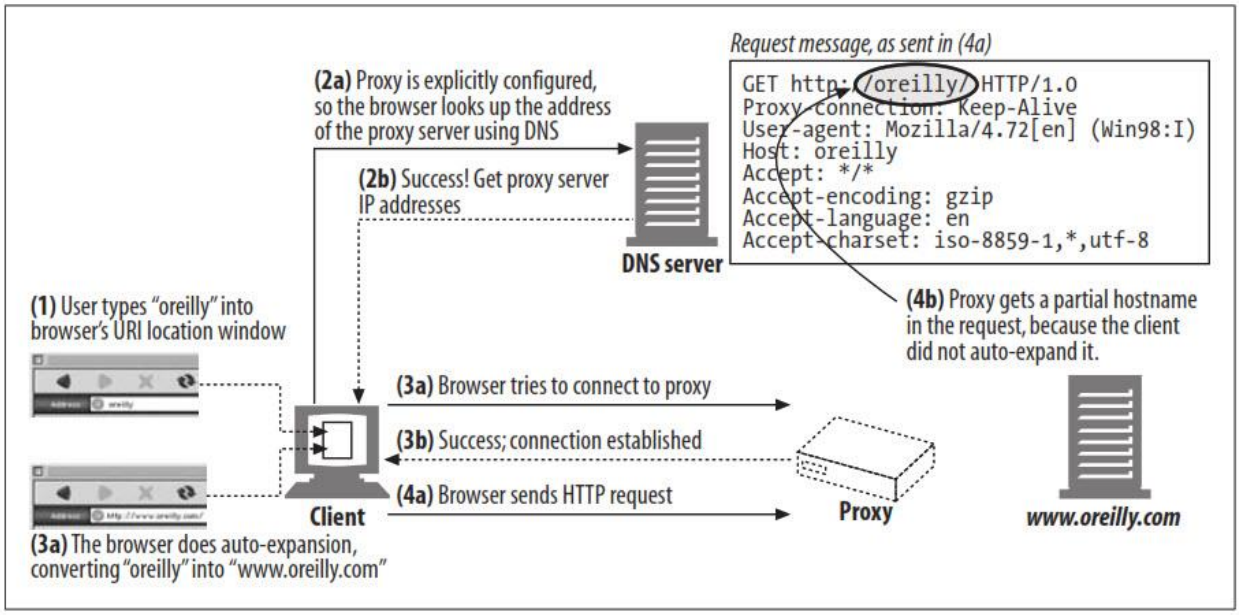
این چیزی است که در این شکل اتفاق می‌افتد:

- در مرحله ۱، کاربر "oreilly" را در پنجره URI مرورگر تایپ می‌کند. مرورگر از "oreilly" به عنوان نام میزبان استفاده می‌کند و یک طرح پیش فرض "http://"، یک پورت پیش فرض "۸۰" و یک مسیر پیش فرض "/" را برای آن در نظر می‌گیرد.
- در مرحله 2a، مرورگر میزبان "oreilly" را جستجو می‌کند. این مورد شکست می‌خورد.
- در مرحله 3a، مرورگر به طور خودکار نام میزبان را گسترش می‌دهد و از DNS می‌خواهد که "www.oreilly.com" را Resolve کند. این بخش موفقیت آمیز است.
- سپس مرورگر با موفقیت به www.oreilly.com متصل می‌شود.

URI Resolution with an Explicit Proxy

وقتی از یک Explicit Proxy استفاده می‌کنید، مرورگر دیگر هیچ یک از این بسط‌های(Expansions) راحت را انجام نمی‌دهد، زیرا URI کاربر مستقیماً به پروکسی ارسال می‌شود.

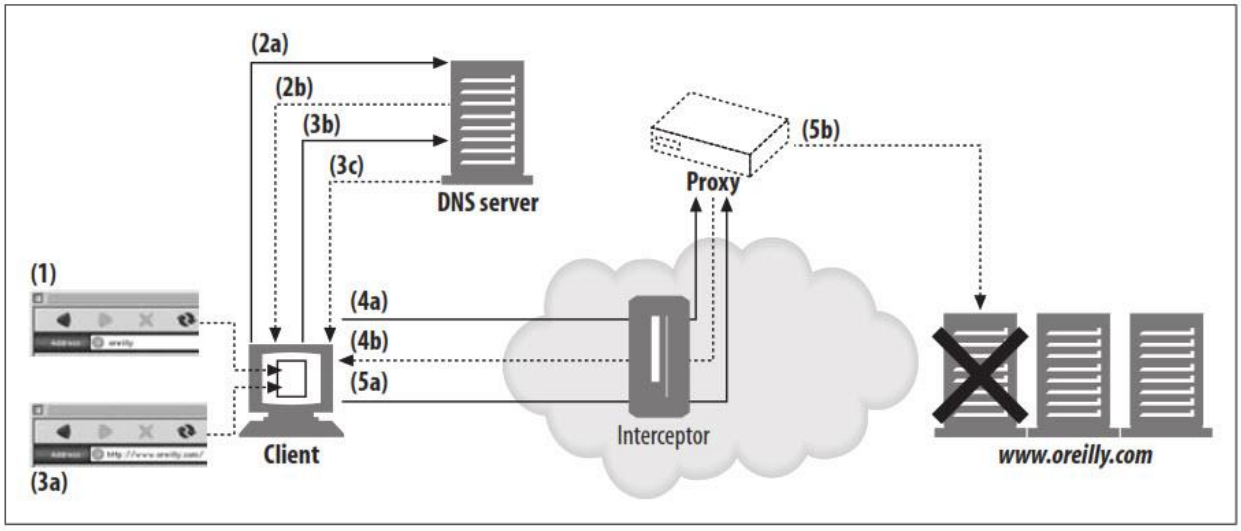
همانطور که در شکل زیر نشان داده شده است، وقتی یک Explicit Proxy وجود دارد، مرورگر نام میزبان جزئی را به طور خودکار گسترش نمی‌دهد. در نتیجه، زمانی که کاربر «oreilly» را در پنجره مکان مرورگر تایپ می‌کند، پروکسی «http://oreilly/» را ارسال می‌کند (مرورگر طرح و مسیر پیش‌فرض را اضافه می‌کند اما نام میزبان را همانطور که وارد کرده‌اید باقی می‌گذارد).



به همین دلیل، برخی از پروکسی‌ها تلاش می‌کنند تا حد امکان از خدمات راحتی مرورگر، از جمله گسترش خودکار «www...com» و افزودن پسوندهای دامنه محلی، تقلید کنند.

URI Resolution with an Intercepting Proxy

Hostname Resolution با یک Intercepting Proxy نامرئی کمی متفاوت است، زیرا تا آنجا که به کلاینت مربوط می‌شود، هیچ پروکسی وجود ندارد! این رفتار تقریباً مانند مورد سرور پیش می‌رود و مرورگر به طور خودکار نام میزبان را تا موفقیت DNS گسترش می‌دهد. اما همانطور که شکل زیر نشان می‌دهد، زمانی که اتصال به سرور برقرار می‌شود، تفاوت قابل توجهی رخ می‌دهد.



شکل بالا تراکنش زیر را نشان می‌دهد:

در مرحله ۱، کاربر "oreilly" را در پنجره URI Location مرورگر تایپ می‌کند.

در مرحله 2a، مرورگر، میزبان "oreilly" از طریق DNS جستجو می‌کند، اما سرور DNS از کار می‌افتد و پاسخ می‌دهد که میزبان ناشناخته است، همانطور که در مرحله 2b نشان داده شده است.

در مرحله 3a، مرورگر توسعه خودکار را انجام می‌دهد و "oreilly" را به ".www. oreilly.com" تبدیل می‌کند. در مرحله 3b، مرورگر، میزبان "www.oreilly.com" را از طریق DNS جستجو می‌کند. این بار، همانطور که در مرحله 3c نشان داده شده است، سرور DNS موفق است و آدرس‌های IP را به مرورگر برمی‌گرداند.

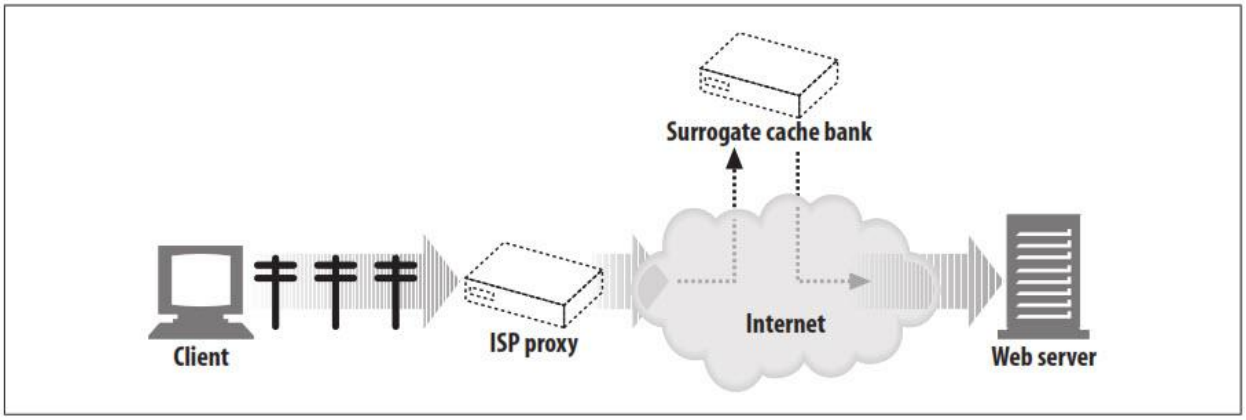
در مرحله 4a، کلاینت قبلاً نام میزبان را با موفقیت Resolve کرده است و لیستی از آدرس‌های IP را دارد. به طور معمول، کلاینت سعی می‌کند به هر آدرس IP متصل شود تا زمانی که موفق شود، زیرا ممکن است برخی از آدرس‌های IP از بین رفته باشند. اما با یک Intercepting Proxy، اولین تلاش برای اتصال توسط سرور پروکسی خاتمه می‌یابد، نه سرور مبدا. در این حالت کلاینت معتقد است که با موفقیت با سرور وب صحبت می‌کند، اما وب سرور حتی ممکن است Alive نباشد.

هنگامی که پروکسی در نهایت آماده تعامل با سرور اصلی واقعی است (مرحله 5b)، پروکسی ممکن است متوجه شود که آدرس IP در واقع به یک سرور پایین اشاره دارد. برای ارائه همان سطح تحمل خطای ارائه شده توسط مرورگر، پروکسی باید آدرس‌های IP دیگر را امتحان کند، این کار یا با Resolve مجدد نام میزبان در هدر میزبان انجام شده و یا با انجام یک جستجوی معکوس DNS در آدرس IP صورت می پذیرد. مهم است که هر دو اجرای Intercepting Proxy و Explicit Proxy از تحمل خطا در DNS

Resolotion به سرورهای مرده پشتیبانی کنند، زیرا وقتی مرورگرها برای استفاده از یک Explicit Proxy پیکربندی می‌شوند، برای تحمل خطا به پروکسی متکی هستند.

Tracing Messages

امروزه، غیرمعمول نیست که درخواست‌های وب از طریق زنجیره‌ای از دو یا چند پروکسی در مسیر خود از کلاینت به سرور عبور کنند (شکل زیر). به عنوان مثال بسیاری از شرکت‌ها از سرورهای پروکسی Cache برای دسترسی به اینترنت، برای صرفه جویی در امنیت و هزینه استفاده نموده و بسیاری از ISP های بزرگ از Proxy Cache برای بهبود عملکرد و پیاده سازی ویژگی‌ها استفاده می‌کنند. امروزه درصد قابل توجهی از درخواست‌های وب از طریق پروکسی‌ها انجام می‌شود. در همان زمان، به دلایل عملکردی، Replicate محتوا در بانک‌های Cache پراکنده در سراسر جهان به طور فزاینده‌ای محبوب می‌شود.



پروکسی‌ها توسط فروشندگان مختلف توسعه می‌یابند. آن‌ها دارای ویژگی‌ها و اشکالات مختلفی هستند و توسط سازمان‌های مختلف مدیریت می‌شوند.

همانطور که پروکسی‌ها رایج تر می‌شوند، باید بتوانید جریان پیام‌ها را در میان پروکسی‌ها ردیابی کنید و هر گونه مشکلی را شناسایی کنید، همانطور که ردیابی جریان بسته‌های IP در سوئیچ‌ها و روترهای مختلف مهم است، این موضوع نیز اهمیت دارد.

The Via Header

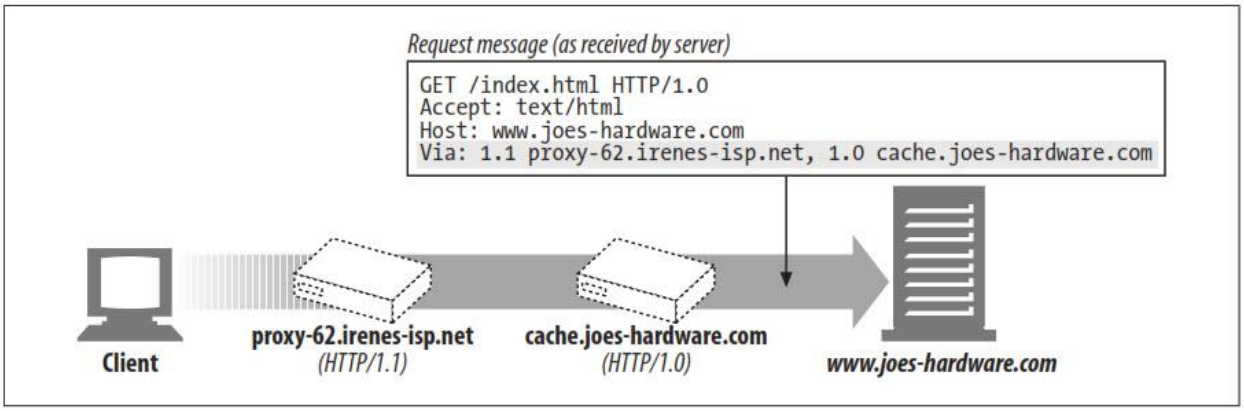
فیلد هدر Via اطلاعات مربوط به هر گره میانی (پروکسی یا Gateway) را فهرست می‌کند که پیام از آن عبور می‌کند. هر بار که پیامی از گره دیگری عبور می‌کند، گره میانی باید به انتهای لیست Via اضافه شود.

رشته Via زیر به ما می‌گوید که پیام از طریق دو پروکسی عبور کرده است. این نشان می‌دهد که پروکسی اول پروتکل HTTP/1.1 را اجرا کرده و proxy-62.irenes-isp.net نامیده می‌شود، و پروکسی دوم HTTP/1.0 را پیاده سازی کرده و cache.joes-hardware.com نامیده می‌شود:

Via: 1.1 proxy-62.irenes-isp.net, 1.0 cache.joes-hardware.com

فیلد Via header برای ردیابی ارسال پیام‌ها، تشخیص حلقه‌های پیام و شناسایی قابلیت‌های پروتکل همه فرستنده‌ها در طول زنجیره درخواست/پاسخ استفاده می‌شود.

پروکسی‌ها همچنین می‌توانند از هدرهای Via برای شناسایی حلقه‌های مسیریابی در شبکه استفاده کنند. یک پروکسی باید قبل از ارسال درخواست، یک رشته منحصر به فرد مرتبط با خودش را در سربرگ Via وارد کند و باید وجود این رشته را در درخواست‌های دریافتی برای شناسایی حلقه‌های مسیریابی در شبکه بررسی نماید.



Via syntax

فیلد هدر Via شامل لیستی از waypoints است که با کاما از هم جدا شده اند. هر waypoint نشان دهنده یک سرور پروکسی یا Gateway هاپ است و حاوی اطلاعاتی در مورد پروتکل و آدرس آن گره میانی است. در اینجا یک مثال از یک هدر Via با دو نقطه بین آمده است:

Via = 1.1 cache.joes-hardware.com, 1.1 proxy.irenes-isp.net

Syntax رسمی هدر Via در اینجا نشان داده شده است:

```
Via               = "Via" ":" 1#( waypoint )
waypoint          = ( received-protocol received-by [ comment ] )
received-protocol = [ protocol-name "/" ] protocol-version
received-by       = ( host [ ":" port ] ) | pseudonym
```

توجه داشته باشید که هر Waypoint در Via دارای حداکثر چهار جزء است: یک نام پروتکل اختیاری (به طور پیش فرض برای HTTP)، یک نسخه پروتکل مورد نیاز، یک نام گره مورد نیاز و یک نظر توصیفی اختیاری:

Protocol name

پروتکل دریافت شده توسط یک واسطه است. اگر پروتکل HTTP باشد، نام پروتکل اختیاری است. در غیر این صورت، نام پروتکل به نسخه اضافه شده و با یک "/" از هم جدا شده است. پروتکل‌های غیر HTTP زمانی رخ می‌دهند که Gateway ها درخواست‌های HTTP را برای پروتکل‌های دیگر (HTTPS، FTP و غیره) متصل می‌کنند.

Protocol version

نسخه پیام دریافت شده فرمت نسخه به پروتکل بستگی دارد. برای HTTP، از شماره نسخه استاندارد استفاده می‌شود ("۱,۰"، "۱,۱"، و غیره). این نسخه در قسمت Via گنجانده شده است، بنابراین برنامه‌های بعدی قابلیت‌های پروتکل تمام واسطه‌های قبلی را خواهند دانست.

Node name

هاست و شماره پورت اختیاری واسطه (اگر پورت گنجانده نشده باشد، می‌توانید پورت پیش فرض پروتکل را در نظر بگیرید). در برخی موارد ممکن است یک سازمان به دلایل حفظ حریم خصوصی، مایل به ارائه نام میزبان واقعی نباشد، در این صورت ممکن است با یک نام مستعار جایگزین شود.

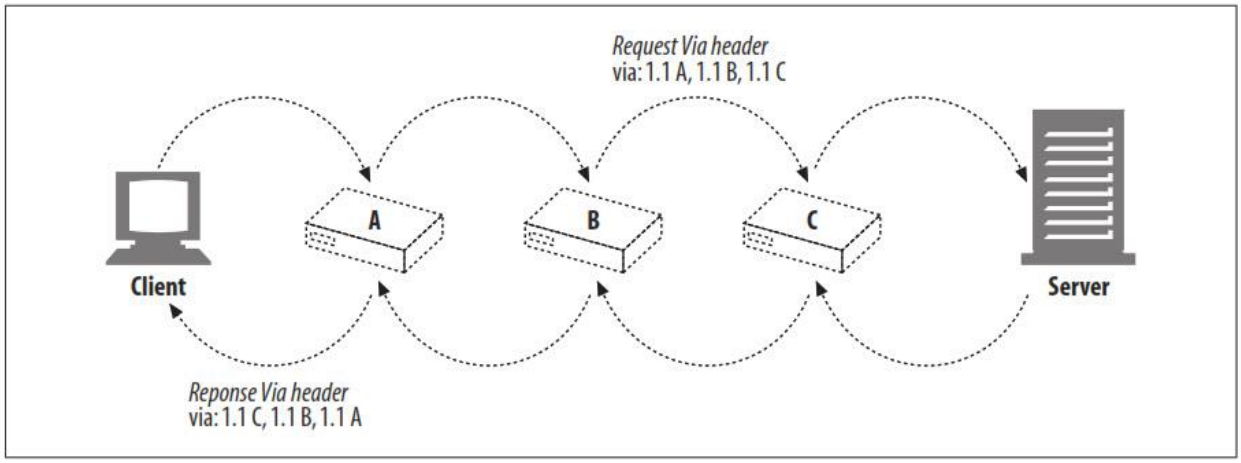
Node comment

یک comment اختیاری که گره واسطه را بیشتر توضیح می‌دهد. معمول است که اطلاعات فروشنده و نسخه را در اینجا درج کنید و برخی از سرورهای پروکسی نیز از فیلد نظر برای درج اطلاعات عیب‌یابی درباره رویدادهایی که در آن دستگاه رخ داده است استفاده می‌کنند.

Via request and response paths

هر دو پیام درخواست و پاسخ از طریق پروکسی‌ها عبور می‌کنند، بنابراین هر دو پیام درخواست و پاسخ دارای هدر Via هستند.

از آنجایی که درخواست‌ها و پاسخ‌ها معمولاً از طریق یک اتصال TCP انجام می‌شوند، پیام‌های پاسخ در همان مسیر درخواست‌ها به عقب حرکت می‌کنند. اگر یک پیام درخواست از طریق پروکسی‌های A، B و C عبور کند، پیام پاسخ مربوطه از طریق پروکسی‌های C، B و سپس A حرکت می‌کند. بنابراین، هدر Via برای پاسخ‌ها تقریبا همیشه برعکس هدر Via برای پاسخ‌ها است (شکل زیر).

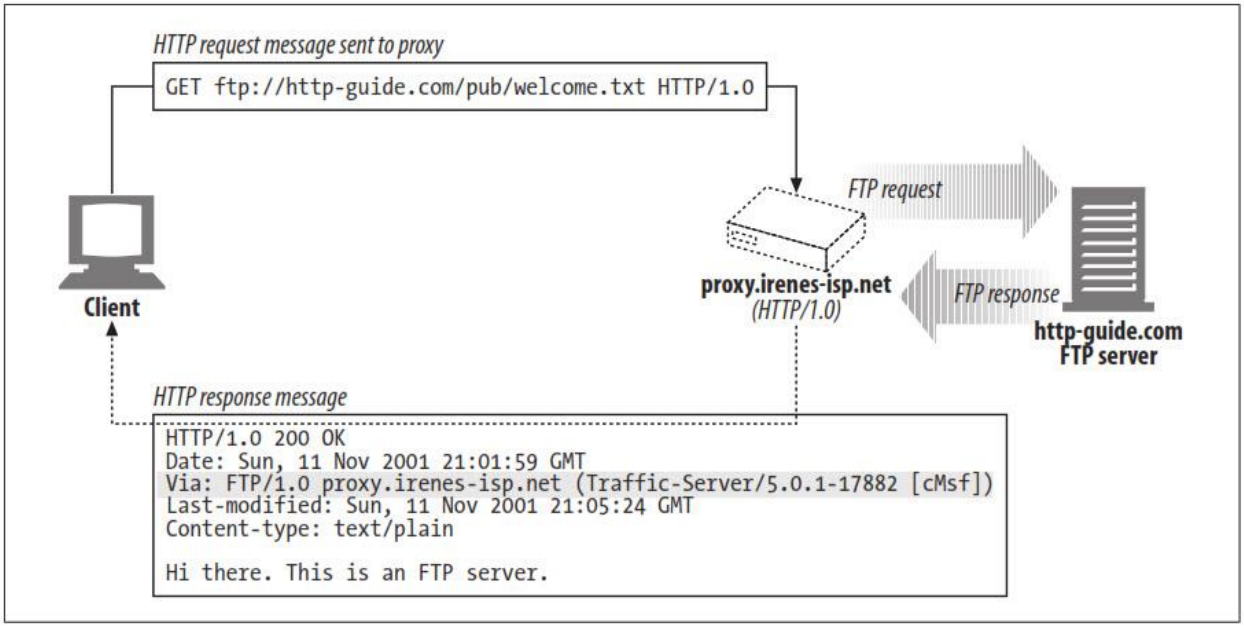


Via and gateways

برخی از پروکسی‌ها عملکرد Gateway را برای سرورهایی که با پروتکل‌های غیر HTTP صحبت می‌کنند ارائه می‌کنند. هدر Via این تبدیل‌های پروتکل را ثبت می‌کند، بنابراین برنامه‌های HTTP می‌توانند از قابلیت‌های پروتکل و تبدیل‌ها در طول زنجیره پروکسی آگاه باشند. شکل زیر یک سرویس گیرنده HTTP را نشان می‌دهد که یک URI FTP را از طریق یک دروازه HTTP/FTP درخواست می‌کند.

کلاینت یک درخواست HTTP برای ftp://http-guide.com/pub/welcome.txt به gateway proxy.irenes-isp.net ارسال می‌کند. پروکسی که به عنوان Gateway پروتکل عمل می‌کند، شی مورد نظر را با استفاده از پروتکل FTP از سرور FTP بازیابی می‌کند. سپس پروکسی با این فیلد هدر Via، شیء را در یک پاسخ HTTP به کلاینت می‌فرستد:

Via: FTP/1.0 proxy.irenes-isp.net (Traffic-Server/5.0.1-17882 [cMs f])



توجه کنید پروتکل دریافتی FTP است. Comment اختیاری شامل نام تجاری و شماره نسخه سرور پروکسی و برخی اطلاعات تشخیصی فروشنده است.

The Server and Via headers

فیلد هدر پاسخ سرور، نرم افزار مورد استفاده توسط سرور مبدا را توصیف می‌کند. در اینجا چند نمونه هستند:

Server: Apache/1.3.14 (Unix) PHP/4.0.4

Server: Netscape-Enterprise/4.1

Server: Microsoft-IIS/5.0

اگر یک پیام پاسخ از طریق یک پروکسی ارسال می‌شود، مطمئن شوید که پروکسی هدر سرور را تغییر نمی‌دهد. هدر Server برای سرور مبدا در نظر گرفته شده است. در عوض، پروکسی باید یک ورودی Via اضافه کند.

Privacy and security implications of Via

مواردی وجود دارد که می‌خواهیم نام‌های میزبان دقیق را در رشته Via وجود نداشته باشد. به طور کلی، مگر اینکه این رفتار به طور صریح فعال باشد، زمانی که یک سرور پروکسی بخشی از فایروال شبکه است، نباید نام و پورت هاست‌ها را در پشت فایروال ارسال کند، زیرا دانش معماری شبکه در پشت فایروال ممکن است برای یک طرف مخرب مفید باشد.(افراد مخرب می‌توانند از نام رایانه‌ها و شماره نسخه‌ها برای آشنایی با معماری شبکه در پشت محیط امنیتی استفاده کنند. این اطلاعات ممکن است در حملات امنیتی مفید باشد. علاوه بر این، نام رایانه‌ها ممکن است سرنخ‌هایی برای پروژه‌های خصوصی در یک سازمان باشد.)

اگر Via node-name forwarding فعال نباشد، پروکسی‌هایی که بخشی از محیط امنیتی هستند باید نام میزبان را با نام مستعار مناسب برای آن میزبان جایگزین کنند. با این حال، به طور کلی، پروکسی‌ها باید سعی کنند برای هر سرور پروکسی یک ورودی Via را حفظ کنند، حتی اگر نام واقعی مبهم باشد.

برای سازمان‌هایی که الزامات حفظ حریم خصوصی بسیار قوی برای پنهان کردن طراحی و توپولوژی معماری‌های شبکه داخلی دارند، یک پروکسی ممکن است دنباله‌ای مرتب از ورودی‌های Via (با مقادیر پروتکل دریافتی یکسان) را در یک ورودی واحد و متصل ترکیب کند. مثلا:

Via: 1.0 foo, 1.1 devirus.company.com, 1.1 access-logger.company.com

می‌تواند جمع شود به:

Via: 1.0 foo, 1.1 concealed-stuff

چندین ورودی را نباید با یکدیگر ترکیب نکنید مگر اینکه همه آن‌ها تحت کنترل سازمانی یکسانی باشند و میزبان‌ها قبلاً با نام مستعار جایگزین شده باشند. همچنین، ورودی‌هایی را که دارای مقادیر پروتکل دریافتی متفاوتی هستند، با یکدیگر ترکیب نکنید.

The TRACE Method

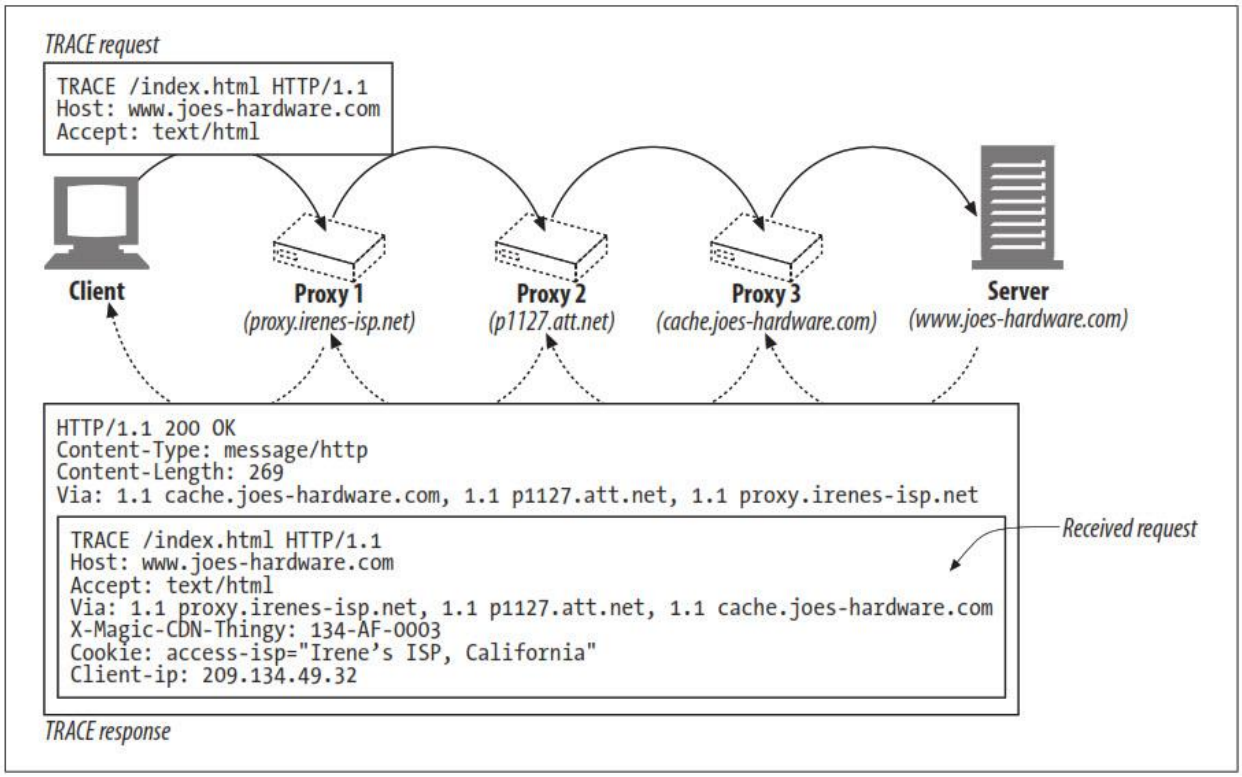
سرورهای پروکسی می‌توانند پیام‌ها را در حین ارسال پیام‌ها تغییر دهند. هدرها اضافه، اصلاح و حذف می‌شوند و بدنه‌ها را می‌توان به فرمت‌های مختلف تبدیل کرد. همانطور که پروکسی‌ها پیچیده‌تر می‌شوند و فروشندگان بیشتری محصولات پروکسی را مستقر می‌کنند، مشکلات قابلیت همکاری افزایش می‌یابد. برای تشخیص آسان شبکه‌های پروکسی، ما به روشی نیاز داریم تا به راحتی شاهد تغییر پیام‌ها در حین ارسال، hop by hop، از طریق شبکه پروکسی HTTP باشیم.

متد TRACE در HTTP/1.1 به شما این امکان را می‌دهد که پیام درخواست را از طریق زنجیره‌ای از پروکسی‌ها ردیابی کنید. همچنین مشاهده کنید که پیام از چه پروکسی‌هایی عبور می‌کند و چگونه هر پروکسی پیام درخواست را تغییر می‌دهد. TRACE برای اشکال زدایی جریان‌های پروکسی بسیار مفید است.

هنگامی که درخواست TRACE به سرور مقصد می‌رسد، کل پیام درخواست به فرستنده منعکس می‌شود و در بدنه یک پاسخ HTTP جمع شده است. هنگامی که پاسخ TRACE می‌رسد، کلاینت می‌تواند دقیقاً پیامی را که سرور دریافت کرده و لیست پروکسی‌هایی که از آن عبور کرده است (در هدر Via) بررسی کند. پاسخ TRACE دارای پیام با Content-Type: message/http و وضعیت 200 OK است.

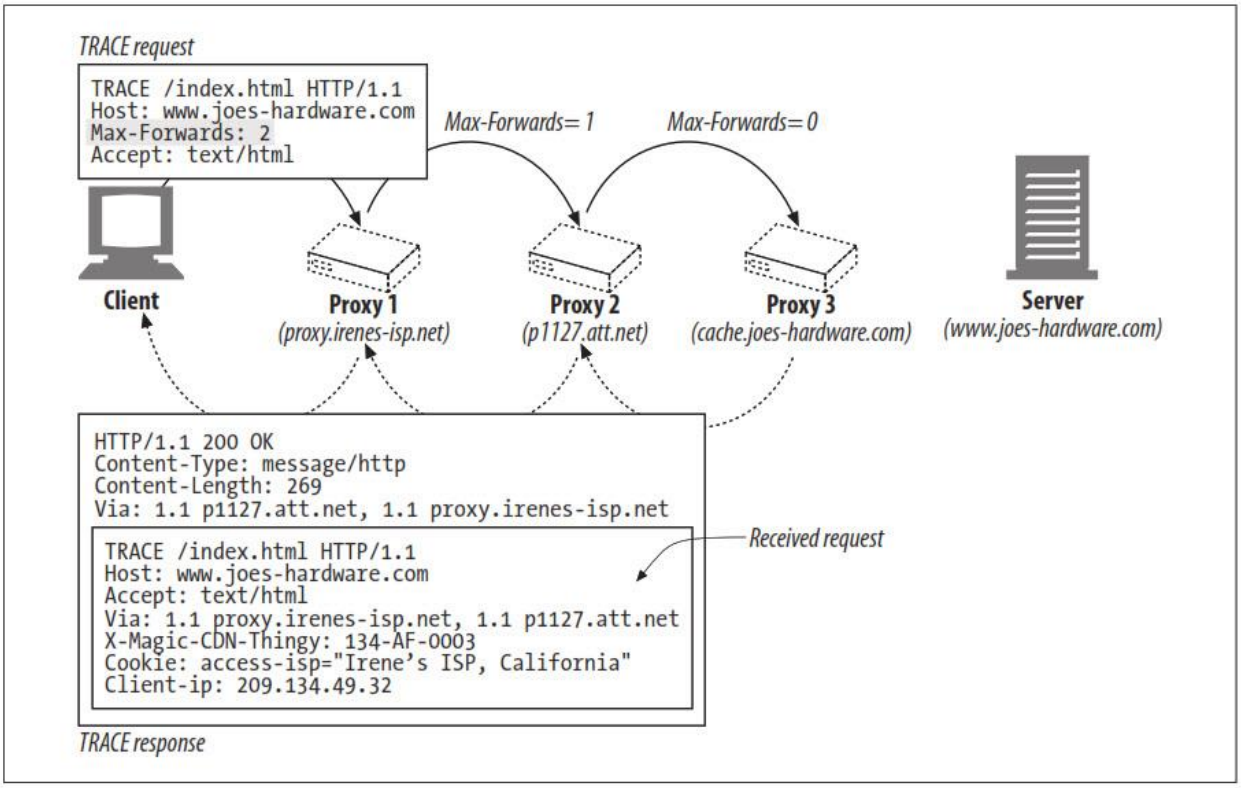
Max-Forwards

به طور معمول، پیام‌های TRACE بدون در نظر گرفتن تعداد پروکسی‌های مداخله‌گر، تمام مسیر را تا سرور مقصد طی می‌کنند. می‌توانید از هدر Max-Forwards برای محدود کردن تعداد پروکسی‌ها برای درخواست‌های TRACE و OPTIONS استفاده کنید، که برای آزمایش زنجیره‌ای از پروکسی‌ها که پیام‌ها را در یک حلقه نامحدود ارسال می‌کنند یا برای بررسی اثرات سرورهای پراکسی خاص در وسط یک زنجیر مفید است. Max-Forwards همچنین ارسال پیام‌های OPTIONS را محدود می‌کند.



فیلد هدر درخواست Max-Forwards حاوی یک عدد صحیح است که تعداد دفعات باقیمانده ارسال این پیام درخواست را نشان می‌دهد. اگر مقدار Max-Forwards صفر باشد (Max-Forwards: 0)، گیرنده باید پیام TRACE را بدون ارسال بیشتر به سمت کلاینت منعکس کند، حتی اگر گیرنده سرور اصلی نباشد.

اگر مقدار Max-Forwards دریافتی بزرگتر از صفر باشد، پیام ارسال شده باید حاوی یک قسمت Max-Forwards به روز شده باشد که مقدار آن یک کاهش یافته است. همه پروکسی‌ها و Gateway ها باید از Max-Forwards پشتیبانی کنند. می‌توانید از Max-Forwards برای مشاهده درخواست در هر hop در یک زنجیره پروکسی استفاده کنید.



Proxy Authentication

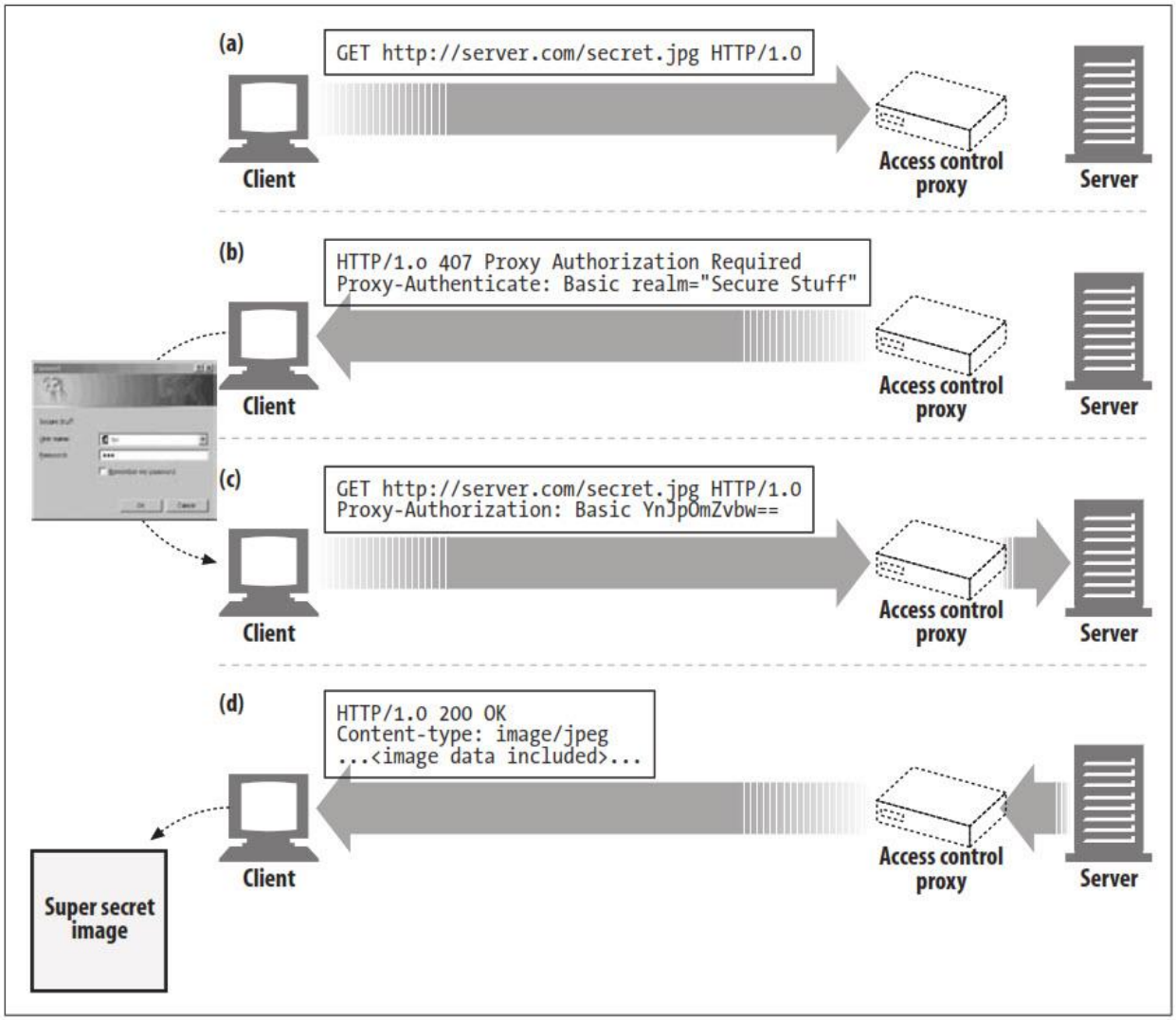
پروکسی‌ها می‌توانند به عنوان دستگاه‌های کنترل دسترسی عمل کنند. HTTP مکانیزمی به نام احراز هویت پروکسی را تعریف می‌کند که درخواست‌های محتوا را تا زمانی که کاربر اعتبار مجوز دسترسی معتبر را به پروکسی ارائه کند، مسدود می‌کند:

هنگامی که درخواستی برای محتوای محدود شده به یک سرور پروکسی می‌رسد، سرور پروکسی می‌تواند یک کد وضعیت 407 Proxy Authorization Required را بازگرداند که خواستار مجوزهای دسترسی است، همراه با یک فیلد هدر Proxy-Authenticate که نحوه ارائه این اعتبارنامه‌ها را توضیح می‌دهد (بخش b شکل).

هنگامی که مشتری پاسخ ۴۰۷ را دریافت می‌کند، سعی می‌کند اعتبار مورد نیاز را از یک پایگاه داده محلی یا با درخواست از کاربر جمع آوری کند.

هنگامی که اعتبارنامه‌ها به دست آمد، کلاینت درخواست را دوباره ارسال می‌کند و اعتبار مورد نیاز را در فیلد هدر Proxy-Authorization ارائه می‌کند.

اگر اعتبارنامه‌ها معتبر باشند، پروکسی درخواست اصلی را در امتداد زنجیره ارسال می‌کند (بخش c شکل). در غیر این صورت یک پاسخ ۴۰۷ دیگر ارسال می‌شود.



احراز هویت پروکسی معمولاً زمانی که چندین پروکسی در یک زنجیره وجود دارد که هر کدام در احراز هویت شرکت می‌کنند، به خوبی کار نمی‌کند. افراد پیشرفت‌هایی را برای HTTP پیشنهاد کرده‌اند تا اعتبارنامه‌های احراز هویت را با waypoint های خاص در یک زنجیره پروکسی مرتبط کنند، اما این پیشرفت‌ها به‌طور گسترده پیاده‌سازی نشده‌اند.

در فصل ۱۲ توضیح دقیق مکانیسم های احراز هویت HTTP، آورده می‌شود.

Proxy Interoperation

کلاینت‌ها، سرورها و پروکسی‌ها توسط چندین فروشنده، با نسخه‌های مختلف مشخصات HTTP ساخته می‌شوند. آن‌ها از ویژگی‌های مختلف پشتیبانی می‌کنند و باگ‌های مختلفی دارند. سرورهای پروکسی

باید بین دستگاه‌های سمت سرویس گیرنده و سمت سرور واسطه شوند، که ممکن است پروتکل@های مختلفی را پیاده سازی کنند و خصلت‌های دردسرساز داشته باشند.

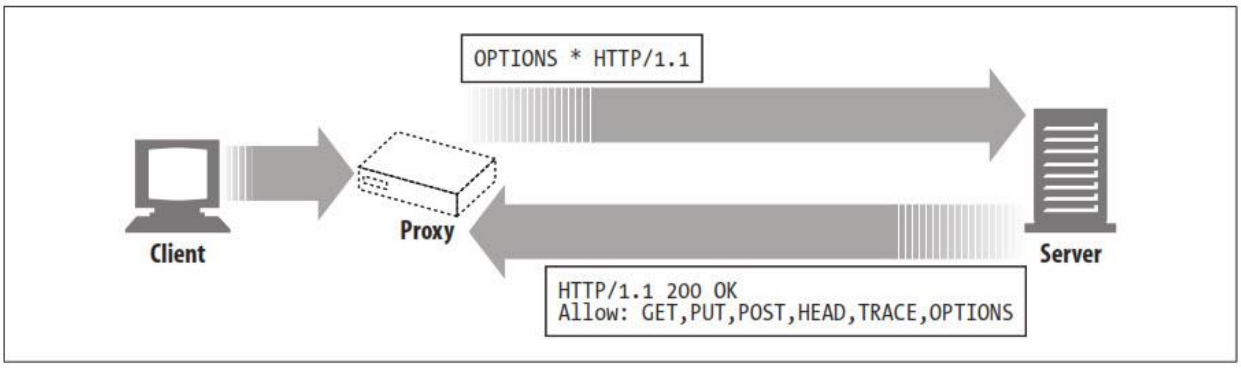
Handling Unsupported Headers and Methods

سرور پروکسی ممکن است تمام فیلدهای هدر را که از آن عبور می‌کنند درک نکند. برخی از هدرها ممکن است جدیدتر از خود پروکسی باشند. برخی دیگر ممکن است فیلدهای هدر سفارشی شده برای یک برنامه خاص باشد. پروکسی‌ها باید فیلدهای هدر ناشناخته را ارسال نموده و باید ترتیب نسبی فیلدهای هدر را با همان نام حفظ کنند.

پروکسی‌هایی که نمی‌توانند متدهای پشتیبانی‌نشده را تونل کنند، ممکن است امروزه در اکثر شبکه‌ها قابل اجرا نباشند، زیرا دسترسی Hotmail از طریق Microsoft Outlook استفاده گسترده‌ای از روش‌های پسوند HTTP می‌کند.

OPTIONS: Discovering Optional Feature Support

متد OPTIONS به یک کلاینت (یا پروکسی) این امکان را می‌دهد تا عملکرد پشتیبانی شده (به عنوان مثال، متدهای پشتیبانی شده) یک وب سرور یا یک منبع خاص در یک وب سرور را شناسایی کند. کلاینت‌ها می‌توانند از OPTIONS برای تعیین قابلیت‌های سرور قبل از تعامل با سرور استفاده کنند و کار با پروکسی‌ها و سرورهای سطوح مختلف ویژگی را آسان‌تر کند.



اگر URI درخواست OPTIONS ستاره (*) باشد، این درخواست به کل عملکرد پشتیبانی شده سرور مربوط می‌شود. مثلا:

```
OPTIONS * HTTP/1.1
```

اگر URI یک منبع واقعی باشد، درخواست OPTIONS در مورد ویژگی‌های موجود برای آن منبع خاص سؤال می‌کند:

OPTIONS http://www.joes-hardware.com/index.html HTTP/1.1

در صورت موفقیت، متد OPTIONS یک پاسخ OK 200 را برمی‌گرداند که شامل فیلدهای هدر مختلف است که ویژگی‌های اختیاری را که در سرور پشتیبانی می‌شوند یا در دسترس منبع هستند، توصیف می‌کنند. تنها فیلد هدری که HTTP/1.1 در پاسخ مشخص می‌کند، هدر Allow است، که توصیف می‌کند چه روش‌هایی توسط سرور (یا منبع خاصی در سرور) پشتیبانی می‌شوند. (همه منابع از هر روشی پشتیبانی نمی‌کنند. به عنوان مثال، یک پرس و جو اسکریپت CGI ممکن است از یک فایل PUT پشتیبانی نکند، و یک فایل HTML ایستا روش POST را نمی پذیرد.)

The Allow Header

فیلد Allow entity header مجموعه ای از متدها را فهرست می‌کند که توسط منبع شناسایی شده توسط URI درخواست یا کل سرور پشتیبانی می‌شود اگر URI درخواست * باشد. مثلا:

Allow: GET, HEAD, PUT

هدر Allow را می‌توان به عنوان سرفصل درخواست برای توصیه روش‌هایی که باید توسط منبع جدید پشتیبانی شوند استفاده کرد. سرور نیازی به پشتیبانی از این متدها ندارد و باید یک عنوان Allow را در پاسخ منطبق گنجانده و متدهای واقعی پشتیبانی شده را فهرست کند.

یک پروکسی نمی‌تواند فیلد Allow header را تغییر دهد، حتی اگر تمام متدهای مشخص‌شده را درک نکند، زیرا ممکن است کلاینت مسیرهای دیگری برای صحبت با سرور اصلی داشته باشد.

For More Information

http://www.w3.org/Protocols/rfc2616/rfc2616.txt

RFC 2616, “Hypertext Transfer Protocol,” by R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Mastinter, P. Leach, and T. Berners-Lee.

http://search.ietf.org/rfc/rfc3040.txt

RFC 3040, “Internet Web Replication and Caching Taxonomy.”

Web Proxy Servers

Ari Luotonen, Prentice Hall Computer Books.

http://search.ietf.org/rfc/rfc3143.txt

RFC 3143, “Known HTTP Proxy/Caching Problems.”

Web Caching

Duane Wessels, O’Reilly & Associates, Inc.