آموزش پراکسی سرور اسکوئید

مرجع آموزشی اسکوئید

سید پوریا حسینی

Porya69info@gmail.com

2012

مهم آن است که هرگز از پرسش باز نایستیم...

آلبرت اینشتین

فهرست مطالب

فصل اول : شروع کار با اسکوئید

فصل دوم : پیکربندی اسکوئید

فصل سوم : راه اندازی اسکوئید

فصل چهارم : به کارگیری ACL ها و Access Rule ها

فصل پنجم : آشنایی با فایلهای ثبت رویداد

فصل ششم : مدیریت اسکوئید و ترافیک شبکه

فصل هفتم : افزایش امنیت شبکه از طریق ابزارهای احراز هویت در اسکوئید

فصل هشتم : راه اندازی و ارتباط گروهی از کش سرورها با یکدیگر

فصل نهم : پیکربندی اسکوئید در حالت پراکسی معکوس

فصل دهم : پیکربندی اسکوئید در حالت شفاف

فصل یازدهم : ابزارهای بازنویسی و تغییر نشانی های URL

فصل دوازدهم : راهنمای رفع اشکال اسکوئید

ضمیمه

مقدمه

سرویس دهنده ی اسکوئید یکی از بهترین و پرکاربردترین نرم افزارهای موجود در سیستم عاملهای لینوکس و یونیکس است که به دلیل ویژگیهای فنی بسیار بالا و متنوع به طور گسترده در مراکز ارائه اینترنت،موسسات آموزشی ، سازمانها و شرکتها مورد استفاده قرار می گیرد. با به کارگیری صحیح و اصولی اسکوئید در شبکه می توان سطح امنیت و بازده شبکه را به طور قابل توجهی افزایش داد.یکی از مشکلات بزرگ در به کارگیری اسکوئید در کشور ما نبود منابع آموزشی جامع و کاربردی است که موجب می شود بسیاری از مدیران شبکه با تنظیمات پیکربندی آن آشنا نبوده و این سرویس دهنده ی قدرتمند را به روشی نادرست در شبکه ی خود پیاده سازی کرده و به جای افزایش کارایی و امنیت باعث ایجاد مشکلات ناخواسته در شبکه شوند.این کتاب به شما کمک می کند این سرویس دهنده را بهتر شناخته و با خصوصیات مختلف آن آشنا شده و مطابق نیازهای و توانایی های خود آن را در شبکه پیاده سازی کنید.در این کتاب فرض بر این بوده است که خواننده هیچ گونه آشنایی با این سرویس دهنده نداشته و تمامی مفاهیم از از پایه شرح داده شده اند.همچنین در طول ارائه ی مطالب مثالهای متنوعی ارائه شده است که هدف از ارائه ی آنها آشنایی هرچه بیشتر با بخشهای مختلف پیکربندی و پیاده سازی آنها در محیط های واقعی است.

سید پوریا حسینی

بهار ۱۳۹۱

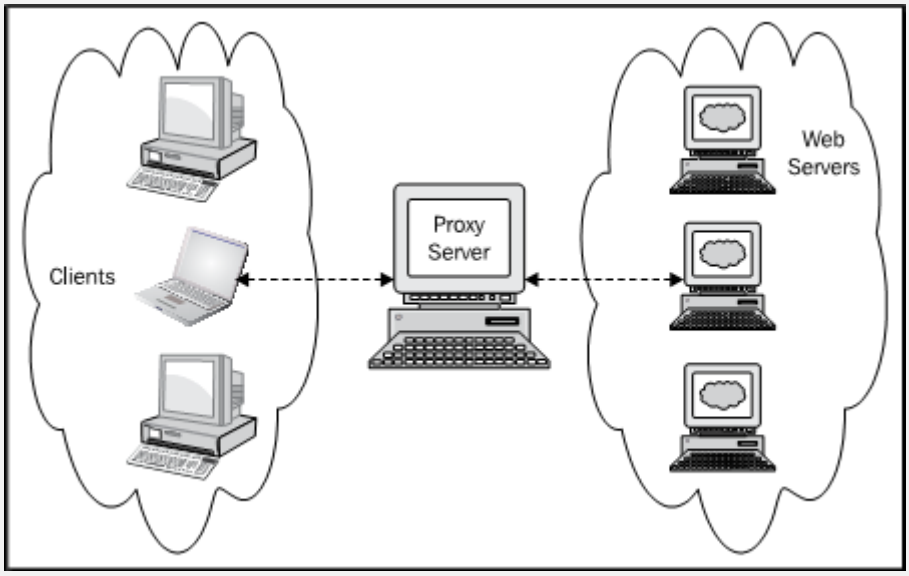
فصل اوّل

شروع کار با اسکوئید

در این فصل به بررسی چگونگی کارکرد پراکسی سرورها در شبکه ،مفهوم کش کردن اطلاعات و همچنین مزایای استفاده از آنها می پردازیم ودر ادامه به معرفی سرویس دهنده اسکوئید،چگونگی دریافت آن،پیش نیازها و راههای نصب آن از طریق سورس کد و بسته های آماده نصب در سیستم عاملهای مختلف و در نهایت مراحل نصب آنها می پردازیم.

پراکسی سرور چیست؟

پراکسی سرور به کامپیوتری گفته می شود که به منظور انجام اعمال خاصی میان درخواستهای کاربران شبکه و سرورهای مقصد که شامل انواع وب سرور ها می شود قرار می گیرد.در ساده ترین حالت یک پراکسی سرور ارتباطات میان کاربران و شبکه خارجی را بدون بررسی و یا پاسخ درخواستها تسهیل می کند،به بیان ساده تر یک پراکسی سرور به مانند یک کاربر ساده درخواستهای کاربران را دریافت کرده و آنها را به سوی سرور های مقصد ارسال می کند و پس از دریافت پاسخ مناسب و قابل قبول آن ها را در اختیار کاربران متقاضی قرار می دهد.در این فرایند یک پراکسی سرور به همان حالتی که وب سرور ها درخواستهای کاربران را دریافت و پاسخ می دهند عمل می کند و یک کاربر معمولی نمی تواند تفاوتی میان این دو قائل نمی شود.در حالت پیچیده تر یک پراکسی سرور پیشرفته قادر به فیلتر کردن محتوا و درخواستهای کاربران ،ایجاد محدودیت در دسترسی ، قابلیت تشخیص هویت برای جلوگیری از ارتباطات ناخواسته و... را داراست و معمولا اساس این اعمال کنترلی از طریق آدرسهای ip و دامنه ها ،انواع پروتکلها و نوع محتوا صورت می گیرد در واقع یک پراکسی سرور با تنظیمات پیشرفته علاوه پاسخگویی به درخواستها قادر به کنترل محتوای مبادله شده میان کاربران و سرورهای مقصد نیز می باشد.



تصویر ۱-۱ موقعیت کلی پراکسی سرور در شبکه را نشان می دهد

با دقت در تصویر بالا متوجه میشوید که هیچکدام از کاربران نمی توانند به طور مستقیم به اینترنت متصل شوند، و هرکدام از کاربران جهت ارتباط با شبکه خارجی حتما باید از سرور پراکسی عبور کنند.در برخی شبکه ها پراکسی سرورها قادر به پاسخگویی به درخواستها و ذخیره کردن داده های دریافت شده توسط کاربران که حاصل دریافت و ارسال داده با سرورهای

خارجی هستند می باشند که به این عمل به اصطلاح cache کردن داده ها گفته می شود.فرایند کش کردن داده ها توسط پراکسی سرورها یکی از محبوبترین ابزار ها به منظور صرفه جویی در مصرف پهنای باند،بهبود مشاهده صفحات وب توسط کاربران و همچنین کاهش زمان بارگذاری صفحات وب است.به طور کلی شکل کار یک پراکسی سرور بدین گونه است که یک کپی از تمامی درخواستهای کاربران را هنگام پاسخگویی در حافظه دیسک سخت ذخیره کرده و برای درخواستهای مشابه بعدی از آنها استفاده می کند ،بدین ترتیب میزان مراجعه به سرورهای خارجی را کاهش می دهد ودر مصرف پهنای باند صرفه جویی می کند.

به طور کلی پراکسی سرورها برای تحقق اهداف زیر به کار می روند:

- کاهش میزان پهنای باند مصرفی
- کاهش زمان بارگذاری صفحات وب با استفاده از قابلیت ذخیره سازی داده ها
- اعمال سیاستهای کنترلی دسترسی به شبکه
- بررسی و کنترل فعالیت های کاربران در قالب گروهها و یا به صورت فردی و همچنین بررسی میزان پهنای باند مصرفی توسط آنها
- افزایش میزان امنیت و حفظ حریم شخصی کاربران که نتیجه ی عبور درخواستهای کاربران پراکسی سرور و درنتیجه مخفی ماندن اطلاعات حساس مانند آدرس IP و...
- قابلیت تقسیم ترافیک شبکه میان وب سرور های مختلف و درنتیجه کاهش بارترافیکی برروی وب سرورها
- بهبود عملکرد وب سرورهای با عملکرد کند در شبکه
- محدودیت در دسترسی به برخی درخواستها و پاسخهای مرتبط با آنها
- قابلیت توزیع بار ترافیکی به منظور تقسیم ترافیک شبکه میان چند پراکسی سرور
- کمک به کارایی بهتر وب سرورها از طریق کش کردن اطلاعات موجود در آنها و انتقال آن به کاربران در صورت درخواست مجدد

به معنای ساده تر پراکسی سرور عاملی است که میان کاربران و شبکه خارجی قرار گرفته و درخواستهای کاربران را از نظر میزان اعتبار داده ها ، سطوح دسترسی و... بررسی می کند. واقع پراکسی سرورها مسئولیت اعتبار سنجی درخواستهای کاربران

را با توجه به سیاستهای کلی اعمال شده توسط مدیر شبکه بر عهده دارد.

پراکسی سرور در حالت معکوس

reverse proxy یا پراکسی معکوس به روشی برای ذخیره سازی درخواستها و پاسخ های دریافت شده از یک وب سرور و ذخیره ی آن در دیسک سخت پراکسی سرور گفته می شود،هدف از این کار بالا بردن کیفیت و سرعت دسترسی کاربران به محتوای موجود برروی وب سرورهای مقصد است. روش کار بدین صورت است که هر کاربری با توجه به داده هایی که در طول ارتباط با وب سرورهای مختلف دریافت و یا ارسال کرده یک کپی از تمامی آنها مثلا یک صفحه وب یا یک تصویر را در دیسک سخت پراکسی سرور نگهداری کرده و به محض اینکه کاربر بعدی همان داده ها را درخواست کند پراکسی سرور وارد عمل شده و درخواست کاربر را به جای استفاده از وب سرورهای خارجی از مخزن داده های ذخیره شده ی خود پاسخ می دهد. میزان زمان نگهداری داده های ذخیره شده از طریق هدر های HTTP (HTTP Headers) دریافت شده از وب سرور های مور نظر کنترل و محاسبه می شوند.با استفاده از هدر های HTTP یک پراکسی سرور می تواند تشخیص دهد که یک داده کش شده توسط پراکسی سرور تا چه زمانی باید از طریق سرور پراکسی در اختیار کاربران متقاضی قرار گیرد و چه زمانی اعتبار داده مورد نظر تمام شده و باید دوباره از طریق وب سرور میزبان آن به روزآوری شود.

به طور کلی فرایند ذخیره سازی داده های وب در موارد زیر کاربرد دارد:

- به منظور کاهش مصرف پهنای باند،بسیاری از داده های موجود در وب سایتهای مختلف مانند کدهای جاوا،CSS انواع تصاویر و ویدئو ها،موسیقی ها و سایر اطلاعات موجود در وب سایتها به ندرت دچار تغییر می شوند واین اطلاعات حجم قابل توجهی از اطلاعات موجود برروی وب سرورهای مختلف را شامل می شود.

- توسط شرکتهای ارائه دهنده اینترنت(ISP)ها، که به منظور کاهش زمان بارگذاری صفحات وب و درنتیجه افزایش سرعت و بازده کاربران هنگام مشاهده صفحات وب برای گروه های مختلف کاربران از جمله کاربران دیال آپ و خطوط پرسرعت به کار می رود.

- کاهش بارترافیکی وب سرورها،وب سرورهایی که میزبانی وب سایتهای پربازدید را برعهده دارند بار ترافیکی زیادی را تحمل می کنند که استفاده از یک پراکسی سرور موجب کاهش مراجعات به وب سرورها و در نتیجه افزایش بازده آن ها می گردد.

آشنایی با اسکوئید

سرویس دهنده اسکوئید (Squid) یکی از بهترین سرویس دهنده های توزیعهای لینوکس و یونیکس محسوب می شود.اولین

نسخه ی آن در سال ۱۹۹۶ منتشر شد اسکوئید علاوه بر ویژگی های یک پراکسی سرور از قابلیت ذخیره سازی داده ها یا کش کردن نیز برخوردار است و یکی از مهمترین ویژگیهای آن محسوب می شود همچنین به دلیل ویژگیهای فنی و مدیریتی خوب و پایداری بسیار بالا می توان آن را در شبکه های کوچک و بزرگ و برای اهداف مختلفی چون کنترل دسترسی کاربران،فیلترکردن محتوا،کش کردن اطلاعات و یا ترکیبی از آنها به کار گرفت.همانطور که میدانید اسکوئید منبع باز بوده و جزءنرم افزارهایی است که تحت مجوز GNU/GPL عرضه می شود بدین معنی که شما برای استفاده و کپی برداری از آن هیچ گونه هزینه ای نمی پردازید . هم اکنون گروههای متعددی از توسعه دهندگان در سراسر جهان به توسعه و نگهداری آن می پردازند همچنین به دلیل پشتیبانی از توزیعهای مختلف و حتی ویندوز مایکروسافت دامنه استفاده از آن بسیار وسیع است همچنین به دلیل نوع ساختار و طراحی خود به منابع سخت افزاری کمتری نسبت به سایر رقبای تجاری خود نیازمند است و به دلیل دارا بودن تمامی امکانات مورد نیاز یک پراکسی و کش سرور از آن می توان در محیط های مختلف کاری و آموزشی مانند دانشگاهها ،سازمانها و مکانهای دیگری که نیازمند مدیریت منسجم کاربران و منابع شبکه هستند استفاده کرد.

دریافت اسکوئید

بسته های نصب اسکوئید در انواع مختلفی قابل دریافت است که شما بسته به میزان توانایی و نیازهای خود می توانید از آنها استفاده کنید ،که شامل بسته های گرد حاوی کد منبع (source code) ، بسته های آماده و از پیش کامپایل شده ی DEB و RPM که به ترتیب در توزیعهای مبتنی بر دبیان و ردهت عرضه می شوند اشاره کرد. بسته های نصب اسکوئید همواره در پایگاه رسمی پروژه به آدرس www.squid-cache.org قابل دریافت هستند. همچنین این بسته ها تقریبا در تمامی توزیعهای مبتنی لینوکس و یونیکس پشتیبانی و عرضه می شوند. بسته های کد منبع اسکوئید در وب سایت رسمی آن به صورت پایدار (stable) ، آزمایشی (Beta) ، در حال توسعه (Development) ، و قدیمی (old) قابل دریافت است. در صورتیکه بخواهید از اسکوئید به منظور اهداف تجاری و حساس استفاده کنید حتما از بسته های پایدار استفاده کنید اما در صورتیکه هدف شما آزمایش ویژگیهای جدید و پیدا کردن باگ های نرم افزار است می توانید از بسته های آزمایشی و در حال توسعه نیز استفاده کنید،از بسته های قدیمی تنها زمانی که سرور شما امکان بروزرسانی و توانایی کامپایل بسته های جدید را نداشته باشد استفاده کنید زیرا همواره نسبت به نسخه های بروزرسانی شده آن دارای ویژگی های کمتر و بعضا باگ های و اشکالات برطرف نشده می باشد. با دقت در نحوه نام گذاری بسته ها می توان دریافت که برخی از بسته ها با شماره های ۳،۳.۱،۲.۷،۲.۶،۳.۲ شروع میشوند که اینها در واقع زمانی که اسکوئید از یک نسخه به نسخه دیگر ارتقا پیدا می کند یک شماره جدید به آن اختصاص می باید و بقیه شماره ها مثلا نسخه ۳.۱.۱۲ عدد ۱۲ به معنای نسخه های زیرمجموعه ۳.۱ است که معمولا همراه با رفع اشکال و اصلاح کدهای همان نسخه توسعه می یابد.درواقع نسخه های مختلف اسکوئید به صورت موازی توسعه

می یابند و مهمترین تفاوتی که دارند این است که ویژگیهای ارائه شده در نسخه های جدید با ویژگیهای موجود در نسخه های قدیمی برابر نیستند و همچنین تغییرات نسبتا وسیعی در تنظیمات و دستورات پیکربندی هریک از نسخه ها ایجاد می شود.

با کلیک برروی هر کدام از نسخه های موجود در سایت صفحه ای دیگرحاوی نسخه های زیرمجموعه ی یک نسخه اصلی نمایان می شود که شما می توانید بسته به نیاز خود یکی از آنها را انتخاب کنید همچنین در جدول روبروی آنها فرمت های مختلف دریافت هر بسته مشخص شده است که به ترتیب عبارتند از:tar.gz , tar.bz2 تنها تفاوتی که میان آنها می توان قائل شد کمتر بودن حجم فرمت tar.bz2 نسبت tar.gz است .تصویر زیر صفحه اصلی دریافت بسته های پیدار اسکوئید را نشان می دهد.همواره می توانید جهت مشاهده و دریافت تمامی نسخه های موجود اسکوئید به آدرس http://www.squid-cache.org/Versions مراجعه نمائید.

Stable Versions:

Current recommended versions meant for production use.

Version	First Production Release Date	Latest Release	Latest Release Date
3.1	29 Mar 2010	3.1.10	22 Dec 2010
2.7	27 May 2008	STABLE9	17 Mar 2010
langpack	18 Sep 2008	N/A (dated)	today

تصویر ۱-۲ نسخه های مختلف پایدار اسکوئید که برای اهداف تجاری توصیه می شود را نشان می دهد.

با کلیک برروی هر یک از نسخه های پایدار موجود صفحه ای دیگرکه حاوی تمامی نسخه های زیرمجموعه نسخه اصلی است نمایش داده می شود.

Squid version 3.1

Release	Date	diff	Download
Latest 3.1 series release			
squid-3.1.10	22 Dec 2010		tar.gz (sig) / tar.bz2 (sig) / rsync
See langpack for latest Language Package			
Daily auto-generated release. This is the most recent bug-fixed update to the formal release. see Change details for the fixes included in this bundle.			
squid-3.1.10-20110106	Jan 6 2011		tar.gz(md5) / tar.bz2(md5) / rsync
Squid-3.1 BZR			Launchpad Mirror
Older Releases			
squid-3.1.9	Oct 15, 2010	diff (sig)	tar.gz (sig) / tar.bz2 (sig)
squid-3.1.8	Sep 04, 2010	diff (sig)	tar.gz (sig) / tar.bz2 (sig)

تصویر ۱-۳ نسخه های زیر مجموعه ی یک نسخه ی اصلی را نشان می دهد.

در تصویر ۱-۳ جدولی که حاوی نسخه های ارائه شده برای هر نسخه ی اصلی ، علاوه بر تاریخ عرضه آن در قسمت Download بسته های مختلف دریافت آن که تنها تفاوت آنها در سیستم گرد سازی آنها است به چشم می خورد که با کلیک برروی هر کدام از آنها بسته ی حاوی کدهای منبع نصب اسکوئید قابل دریافت است.

همانطور که قبلا هم اشاره شد تفاوت نسخه های مختلف در این است که تمامی ویژگیهایی که در نسخه ۳ موجود است ممکن است در نسخه ۲.۷ و یا پایین تر یافت نشود و یا برخی از امکانات قدیمی با ویژگی های جدید جایگزین شده باشند به علاوه فایل پیکربندی اصلی اسکوئید و همچنین دستورات نسخه های مختلف با تغییراتی مواجه می شوند اما به طور کلی همواره توصیه می شود از آخرین نسخه ی پایدار اسکوئید استفاده شود.

نصب اسکوئید

سرویس دهنده اسکوئید را می توان از طریق بسته های کد منبع و یا بسته های باینری و از پیش کامپایل شده نصب کرد در ادامه به بررسی نصب آن از هر دو روش مذکور می پردازیم:

نصب از طریق کد منبع (source code)

فرایند نصب اسکوئید از طریق کد منبع به سه مرحله ی کلی زیر تقسیم می شود:

۱. آماده سازی سیستم عامل و نصب کامپایلرهای مورد نیاز و انتخاب ویژگیهای مورد نظر.

۲. کامپایل کد منبع به منظور تولید فایلهای اجرایی.

۳. قرار گیری فایلهای اجرایی ایجاد شده و سایر فایلهای مورد نیاز در مسیرهای مخصوص به خود جهت اجرای صحیح بدون اشکال اسکوئید.

کامپایل اسکوئید

کامپایل اسکوئید شامل کامپایل تعداد زیادی فایل حاوی کدهای Cو++ C است ،فرایند کامپایل اسکوئید بسیار ساده است و پیچیدگی چندانی در آن وجود ندارد .برای شروع کار شما باید کامپایلر های GCC و G++ را برروی سیستم عامل خود نصب کنید.البته در اکثر توزیعهای لینوکس و بونیکس این کامپایلرها به صورت پیش گزیده وجود دارند و نیازی به اضافه کردن آنها نیست.

چرا کامپایل؟

شاید این سوال برای شما هم پیش بیاید که واقعا چرا باید به جای استفاده از بسته های آماده ی اسکوئید که به راحتی و بدون طی مراحل طولانی نصب می شوند باید از گزینه کامپایل استفاده کنیم؟ اما باید بدانید کامپایل کردن مزایایی دارد که بسته های آماده از وجود آنها بی بهره اند از جمله:

- هنگام کامپایل می توان ویژگیهایی را فعال کرد که این ویژگی ها به طور پیش فرض در بسته های آماده و از پیش کامپایل شده غیر فعال هستند.
- هنگام کامپایل می توان ویژگیهایی را که به دلایلی مایل به فعال بودن آنها نیستیم را غیر فعال کنیم به طور مثال می توان ابزارهای احراز هویت را غیرفعال کرد.
- با استفاده از گزینه configure می توان بر تمامی قابلیت هایی که مایل به فعال و غیر فعال کردن آنها هستیم کنترل کامل داشته باشیم این در حالیست که در بسته های آماده تمامی ویژگیهای موجود در آن ثابت هستند و قابلیت حذف و یا اضافه شدن را ندارند.
- در حالت کامپایل می توان محل نصب فایلهای اسکوئید را از مسیر پیش فرض به مسیر مورد نظرتغییر داد همچنین می توان برای اسکوئید مجوزهای لازم جهت اجرا در سایر حساب های کاربری غیر از حساب ریشه(root) را تعریف کرد.به طور کلی نصب اسکوئید از طریق کامپایل مزایای زیادی دارد که بسته های آماده از وجود آن بی بهره اند در مقابل بسته های از پیش کامپایل شده نیز دارای مزایای خاص خود هستند که یکی از مهمترین مزیت آنها نصب ساده و سریع و بدون نیاز به هیچ گونه کامپایل دستی کدها است.

باز کردن بسته های کد منبع

در صورتیکه برای نصب از کد منبع اسکوئید استفاده می کنید ، برای شروع مراحل نصب ابتدا باید آن را از حالت فشرده خارج کنید. به مثال زیر توجه کنید:

```
tar –xvzf  squid-3.1.16 tar.gz
```

دستور tar یکی از پرکاربردترین دستورات برای خارج کردن بسته ها از حالت فشرده است.در مثال بالا از دستور tar -xvfz و نام بسته مورد نظر یعنی squid-3.1.16 tar.gz استفاده شده است که طبیعتا با توجه به زمان و نسخه بسته ی دریافتی توسط شما این نام متفاوت خواهد بود و شما باید نام بسته ی دریافت شده را به جای نامی که دراین مثال آمده است وارد نمائید.

فرمان Configure

configure یا system check اولین مرحله از پروسه ی کامپایل اسکوئید است که با دستور configure/. آغار می شود،این ابزار سیستم عامل را در مواردی چون نصب کتابخانه ها و کامپایلرهای مورد نیاز و همچنین فایل های توصیف گر(file descriptor) و بدست آوردن اطلاعاتی چون معماری سیستم و سایر موارد مورد نیاز بررسی می کند و در صورتیکه یکی از موارد مورد نیاز موجود نباشد و یا با اشکال روبرو باشد یک پیغام خطا در مقابل هر مورد چاپ می کند و در انتها نیز عملیات کامپایل را متوقف کرده و تا زمانی که موارد مورد نیاز فراهم نشوند اجازه کامپایل را نخواهد داد اما در صورتیکه تمامی پیش نیازها وجود داشته باشد پس از پایان این مرحله فایلهای اجرایی لازم را تولید کرده و شرایط را برای آغاز مرحله ی بعدی کامپایل مهیا می کند.به کارگیری دستور configure بدون هیچ پسوندی تمامی تنظیمات و ویژگی های نصب اسکوئید را در حالت پیش فرض قرار می دهد و فقط گزینه هایی که به صورت پیش فرض فعال هستند را آماده کامپایل می کند اما در صورتیکه بخواهیم در مرحله کامپایل ویژگیهایی را فعال و یا غیر فعال کنیم باید از پسوندهای مخصوص هر ویژگی استفاده کنیم و برای مشاهده تمامی گزینه های موجود در این مرحله باید از دستور ./configure --help | less استفاده کنیم.خروجی این دستور نمایش تمامی گزینه ها و پسوندهای مجاز برای استفاده در این مرحله همراه با توضیحی مختصر در هر مورد می باشد که در ادامه با مهمترین این دستورها بیشتر آشنا می شویم.

--prefix

prefix یکی از معمول ترین گزینه هاست و از آن برای مواردی که به تست و بررسی چند نسخه اسکوئید به طور همزمان برروی یک ماشین می پردازیم استفاده می شود. با استفاده از این دستور می توان مسیر دایرکتوری های هر کدام از نسخه های مختلف اسکوئید را تعیین کرد و این کار برای جلوگیری از تداخل نسخه های مختلف با هم ضروری است. نحوه به کارگیری آن بدین صورت است:

./configure –prefix=/opt/squid/3.1.10/

در این مثال نسخه ۳.۱.۱۰ اسکوئید در مسیر opt/squid/3.1.10/ نصب خواهد شد به طور مشابه برای نصب اسکوئید نسخه ۳.۱.۵ به صورت opt/squid/3.1.5/ عمل می شود.

*نکته: از این پس در این کتاب هرگاه در ابتدای دستورها پیشوند {**prefix**} به کار رود ، بدین معنی است که شما باید به جای این عبارت مسیر اصلی فایل مورد نظر اسکوئید را که هنگام کامپایل تعیین کرده اید جایگزین نمائید.*

سرویس دهنده اسکوئید جهت کنترل فایلهای راه انداز خود از گزینه های متعددی به منظور تعیین مسیر فایلهای اجرایی خود

استفاده می کند مثلا گزینه های--sbindir , --bindir بدین منظور ایجاد شده اند .برای فعال کردن یک قابلیت از ساختار کلی --enable-FEATURE_NAME استفاده می شود که به جای FEATURE_NAME نام ویژگی مورد نظر قرارمی گیرد،همچنین برای غیر فعال کردن یک ویژگی از دستورات --disable-FEATURE_NAME و --enableFEATURE_NAME=no استفاده می شود . برای روشن شدن این موضوع به مثال زیر توجه کنید.

./configure --enable-icmp # FEATURE will be enabled

./configure --disable-icmp # FEATURE will be disabled

./configure --enable-icmp=no # FEATURE will be disabled

در مثال بالا در خط اول پروتکل icmp فعال و در خط دوم و سوم غیر فعال شده است تمامی ویژگیهای دیگر نیز به همین شکل مدیریت می شوند.

--enable-gnuregex

در صورتیکه سیستم عامل مورد استفاده شما قدیمی است و نیازمند استفاده از ACL ها هستید فعال کردن این گزینه الزامی است اما درصورتیکه سیستم عامل شما به روز است نیازی به استفاده از این گزینه نیست.

--disable-inline

بسته های نصب اسکوئید حاوی مجموعه کد هایی است که می توان زمان کامپایل از آنها صرف نظر کرد و به طور عمده شامل زبانهای کشورهای مختلف است که برای نمایش پیام های خطای اسکوئید به کار می رود. نصب این کدها زمانی مفید خواهند بود که از اسکوئید برای موارد مهمی مانند موارد تجاری استفاده کرد اما در صورتیکه شما در حال تست و بررسی نسخه های مختلف اسکوئید هستید نیازی به کامپایل این کد ها ندارید و بدین ترتیب زمان مورد نیاز برای کامپایل کاهش می یابد.

--disable-optimizations

اسکوئید به صورت پیش فرض از کامپایلرهای بهینه شده ای استفاده می کند که هنگام کامپایل نتایج و بازده بهتری را از خود نشان می دهند.این دستور نیز در صورت استفاده هنگام تست و یا رفع اشکال زمان مورد نیاز جهت انجام این فرایندها را

کاهش می دهد همچنین در صورت به کارگیری این دستور ، فرمان disable-inline-- نیز به طور خودکار فعال می شود.

--enable-storeio

کارایی اسکوئید خصوصا زمانی که قابلیت ذخیره سازی داده هادر آن فعال باشد به میزان بسیار زیادی به کارایی و سرعت دیسک سخت و نوع سیستم فایل مورد استفاده جهت عملیات خواندن و نوشتن داده ها(I/O) بستگی دارد در واقع هرچه زمان خواندن و نوشتن در دیسک سخت کمتر باشد زمان پاسخگویی به درخواستها کاهش می باید. با استفاده از دستور-enable storeio-- می توان نوع سیستم فایلهای مخصوصی که اسکوئید از آنها برای نگهدار ی داده ها در مخرن خود استفاده می کند را تعیین کرد به طور مثال کد ./configure –enable-storeio=ufs,aufs,coss,diskd,null.سیستم فایل های ufs , aufs ,coss ,diskd , null را فعال می کند.جهت کسب آگاهی از انواع سیستم فایلهای قابل پشتیبانی در نسخه اسکوئید مورد استفاده پوشه ی src/fs/ واقع در پوشه اصلی نصب اسکوئید را بررسی نمائید.

--enable-removal-policies

هنگام استفاده از ویژگی ذخیره سازی داده ها در یک بازه زمانی مشخص ،میزان حافظه اختصاص داده شده به مخزن کش سرور تمام می شود ، در حالیکه درخواستهای کاربران و مطالب جدید تمامی ندارند و اسکوئید به طور مداوم به فضای آزاد برای ذخیره سازی اطلاعات نیاز دارد ، برای حل این مشکل اسکوئید از یک راه حل در کنار اضافه کردن حافظه فیزیکی بهره می برد و آن هم حذف داده های قدیمی در مخزن کش با ساتفاده از یک مکانیسم مشخص است که می تواند در یک بازه زمانی معین داده های قدیمی ای راکه کاربران دیگر به آنها نیازی ندارند را حذف و فضای دیسک سخت را برای ذخیره داده های جدید آزاد نماید . بدین منظور دو راهکار کلی heap و lru وجود دارد. برای فعال کردن این قابلیت فرمان ./configure –enable-removal- policies=heap,lru به کار می رود.

--enable-icmp

اسکوئید از این ابزار به منظور تعیین میزان تاخیر و فاصله کش سرورهای همسایه و همکار و همچنین سرورهای راه دور استفاده می کند.به کارگیری این دستور فقط زمانی مفید است که اسکوئید در یک شبکه با سایر کش سرورهای در ارتباط باشد .در غیر اینصورت فعال و یا غیر فعال بودن آن تاثیری بر عملکرد اسکوئید نخواهد داشت.

--enable-delay-pools

delay-pool روشی برای کنترل مستقیم پهنای باند مصرفی توسط کاربران به وسیله اسکوئید است. فرض کنید شما ۱۰۰ کاربر دائمی دارید که روزانه از شبکه شما استفاده می کنند و شما می خواهید میزان پهنای موجود را به طور مساوی میان همه آنها تقسیم کرده تا کاربران نتوانند بیشتر از یک سرعت مشخص به دریافت و ارسال دادها بپردازند. برای فعال کردن این ویژگی از دستور enable-delay-pools-- استفاده می شود.

--enable-esi

این دستور زمانی به کار می رود که از اسکوئید به عنوان شتاب دهنده به همراه سایر وب سرورهای شبکه به کار رود.در صورت فعال شدن این دستور اسکوئید به طور کلی تمامی هدر های cache-control را از سمت کاربران نادیده می گیرد.همچنین جهت کسب اطلاعات بیشتر در را بطه با این موضوع به نشانی http://www.esi.org مراجعه نمائید.

--enable-useragent-log

این گزینه موجب ایجاد سازگاری بهتر در ثبت رخدادهای مربوط به هدرهای موجود در درخواستهای HTTP ارسال شده از سوی کاربران می شود.

--enable-referer-log

درصورتیکه این ویژگی فعال شود اسکوئید قادر خواهد بود تا ارجاع دهنده هایی را به هدر های HTTP موجود در درخواستهای کاربران هنگام ذخیره آنها در فایل ثبت رخداد اضافه کند.

--disable-wccp

Cisco's Web Cache Communication Protocol (WCCP)یکی از پروتکل های اختصاصی شرکت سیسکو سیستمز است که به منظور برقراری ارتباط سوییچ ها و مسیریابهای این شرکت با کش سرور ها طراحی شده است. این پروتکل به طور پیش فرض فعال است و درصورتیکه تمایل دارید این قابلیت غیر فعال شود از دستورdisable-wccp-- استفاده کنید.

--disable-wccpv2

مشابه مورد قبل wccpv2 نسخه ی ارتقا یافته ی پروتکل wccp است که به آن خصوصیات پیشرفته ای نظیر قابلیت توزیع بار (load balancing)،مقیاس گذاری،مکانیسم های افزایش ضریب اطمینان و تحمل خطا افزوده شده است. برای غیر فعال کردن آن از دستور disable-wccpv2-- استفاده کنید.

--disable-snmp

در نسخه ۳ اسکوئید و بعد از آن پروتکل snmp و یا (simple network management protocol) که یکی از محبوبترین پروتکلهای مدیریت دستگاههای شبکه است به طور پیش گزیده فعال شده است و از آن می توان برای بررسی منابع سرور شامل وضعیت باند مصرفی،وضعیت سخت افزاری سرور وضعیت دیسک سخت و... استفاده کرد.

--enable-cachemgr-hostname

Cache mgr یا (cachemgr.cgi) رابط گرافیکی مدیریت اسکوئید است که به وسیله آن می توان از طریق مرورگر اطلاعات جامعی را در مورد وضعیت کارکرد اسکوئید به دست آورد . به طور پیش گزیده اسکوئید از عبارت localhost برای تعیین نام میزبان خود استفاده می کند،درصورتیکه شما مایل به انتخاب یک نام میزبان متفاوت جهت دسترسی به رابط گرافیکی هستید میتوانید از دستور ./configure –enable-cachemgr-hostname=squidproxy استفاده کنید که در این مثال عبارت squidproxy به عنوان نام میزبان در نظر گرفته شده است.

--enable-arp-acl

اسکوئید علاوه بر توانایی کار با آدرس های ip ،از آدرسهای سخت افزاری کاربران و یا MAC address هم می تواند جهت استفاده در ACL ها استفاده کند.آدرس های MAC درواقع نشانی های سخت افزاری کارتهای شبکه هستند و همانند نشانی های IP ها برای هر کاربر منحصربه فرد هستند. به هر حال در طول یادگیری اسکوئید کار با MAC آدرسها می تواند ایده خوبی برای پیشرفت شما باشد! .

نکته:این دستور با **--enable-eui** *که به صورت پیش فرض فعال است جابجا می شود.*

--disable-htcp

Hypertext Caching Protocol (HTCP) پروتکلی است که اسکوئید از آن برای دریافت و ارسال اطلاعات مربوط به اشیاء ذخیره شده در مخزن کش سرور با سایر کش سرورهای همسایه استفاده می کند و به کارگیری این دستور باعث غیرفعال شدن پشتیبانی از این پروتکل می شود.

--enable-ssl

اسکوئید قادر است ارتباطات نوع SSL را به طور یک جانبه از سمت خود خاتمه دهد.هنگامی که اسکوئید در حالت پراکسی معکوس قرار می تواند درخواستهای کاربرانی را که از نوع SSL است به طور یک طرفه خاتمه دهد و آن را تا میانه ارتباط با وب سرورهای مقصد حمل کند..در این حالت ارتباط میان اسکوئید و وب سرورهای بخش مدیریت از نوع HTTP بدون هیچ گونه رمز نگاری خواهد بود اما کاربران به دلیل قرارگیری اسکوئید میان آنان و سرورهای وقصد قادر به مشاهده وب سایتها از پروتکل SSL هستند.به طور کلی این دستور فقط زمانی مفید است که اسکوئید به عنوان شتاب دهنده و یا پراکسی معکوس با سایر وب سرورها در ارتباط باشد.

--enable-cache-digests

Cache-digests ها یکی دیگر از راهکارهای ارتباطی اسکوئید با سرورهای همسایه خود است که از آن برای ارسال و دریافت اطلاعاتی نظیر مشخصات فایلهای ذخیره شده و سایر اطلاعات به منظور اعلام وضعیت خود به سایر کش سرورها استفاده می کند. در این روش اطلاعات مهمی چون وضعیت اشیاء کش شده به صورت فشرده شده با سایر همسایه ها مبادله می شود.

--enable-default-err-language

هنگامی که درخواستهای کاربران به دلایلی مانند رد درخواست توسط کش سرور ، عدم برقراری ارتباط با سرور مقصد و یا پیدا نشدن صفحه ی مورد نظر با خطا مواجه می شود اسکوئید از یک صفحه خطا شامل اطلاعاتی چون دلیل عدم برقراری ارتباط و راهکارهایی برای رفع این مشکل، نشانی پست الکترونیک مدیر شبکه و همچنین نسخه مورد استفاده اسکوئید و نام میزبان را برای کاربر نمایش می دهد.جهت برقراری ارتباط موثر با کاربران این صفحات به زبانهای رایج دنیا از جمله زبان پارسی ترجمه شده است اما به طور پیش فرض تمامی این صفحات به زبان انگلیسی نمایش داده می شود که برای تغییر زبان پیش فرض می توان از دستور ./configure --enable-default-err-language=persian استفاده کرد.

--enable-err-languages

همانطور که در بالا اشاره شد اسکوئید از بسیاری از زبانهای رایج دنیا به منظور اطلاع رسانی به کاربران در صفحات خطای خود پشتیبانی می کند. درصورتکیه می خواهید تنها از چند زبان خاص پشتیبانی شود می توانید از دستور -- ./configure enable-err-languages='English French German' استفاده کنید.در واقع پس از اضافه کردن این دستور به فهرست گزینه های کامپایل فقط کدهای مربوط به زبانهای انتخابی کامپایل می شوند.جهت آگاهی از فهرست زبانهای قابل پشتیبانی در اسکوئید به پوشه ی errors/ واقع در پوشه اصلی کد منبع اسکوئید مراجعه نمائید.

--disable-http-violations

اسکوئید به طور پیش فرض و با توجه به نوع پیکربندی برخی از استاندارد های عمومی مربوط به پروتکل HTTP را رعایت نمی کندو این کار را به وسیله جابجایی برخی هدر های HTTP انجام می دهد. در صورتکیه مایل به انجام این کار نباشیم و بخواهیم اسکوئید تمامی استاندارد ها را همان گونه که هستند رعایت کند از این دستور استفاده می کنیم.

--enable-ipfw-transparent

IPFIREWALL(IPFW) سیستم دیوار آتش مورد استفاده برروی سیستم عامل های برپایه ی FREEBSD است که توسط توسعه دهندگان و همکاران پروژه FREEBSD پشتیبانی می شود. این دستور زمانی مفید است که از پراکسی سرور بصورت Transparent یا شفاف استفاده شود و در صورتیکه برروی سیستم عامل مورد نظر دیوار آتش IPFW نصب نباشد نباید از این دستور استفاده کرد زیرا در هنگام کامپایل اسکوئید باعث ایجاد خطا خواهد شد. مقدار پیش فرض برای این دستور نوع تشخیص خودکاراست که عمل شناسایی را به خوبی انجام می دهد.

--enable-ipf-transparent

IPFilter یا IPF یکی دیگر از ابزارهای دیوار آتش سیستم عامل های یونیکسی مورد استفاده در توزیع هایی مانند سولاریس و NETBSD است. درصورتیکه ابزار دیوار آتش برروی این سیستم ها از نوع IPF است باید از این دستور استفاده شود در غیر این صورت فعال کردن این گزینه باعث بروز خطا هنگام کامپایل می شود.کاربرد این دستور زمانی است که اسکوئید درحالت شفاف پیکربندی و اجرا شده باشد.

--enable-pf-transparent

Packet Filter (PF)یکی دیگر از ابزارهای دیوار آتش مورد استفاده در سیستم عامل های برپایه ی یونیکس است که به طور اختصاصی برای توزیع Open BSD توسعه داده شده است . مانند موارد بالا درصورتیکه از این سیستم عامل استفاده می کنید و می خواهید از اسکوئید به صورت شفاف استفاده کنید به کار گیری این دستور الزامیست.

--enable-linux-netfliter

netfilter ، یکی دیگر از ابزارهای فیلترینگ و مدیریت بسته های شبکه است که عموما در تمامی توزیعهای لینوکس با هسته ۲.۴ و ۲.۶ و بالاتر پشتیبانی می شود. درصورتی که سیستم عامل میزبان پراکسی سرور یکی از توزیع های لینوکس مانند دبیان،ردهت،فدورا،اوبونتو و...است جهت استفاده از حالت شفاف باید این ویژگی فعال شود .

--enable-follow-x-forwarded-for

هنگامی که یک درخواست نوع HTTP توسط پراکسی سرور ارجاع داده می شود پراکسی سرور اطلاعات مهمی شامل مشخصات خود و کاربر درخواست کننده آن را به هدر درخواست مورد نظر اضافه می کند.اسکوئید از این اطلاعات برای پیدا کردن نشانی ip کاربر اصلی درخواست کننده در میان سایر سرورهای پراکسی استفاده می کند به بیان ساده تر این اطلاعات کاربر اصلی درخواست کننده را از طریق نشانی IP آن پیدا کرده و در صورتکیه درخواست به سایر سرورهای پراکسی نیز ارجاع داده شود باز هم کاربر مورد نظر از طریق این اطلاعات قابل شناسایی خواهد بود.

--disable-ident-lookups

این دستور موجب می شود اسکوئید تعداد اتصالات برقرار شده توسط یک نام کاربری کاربر را کنترل نکند که این خود باعث کاهش سطح امنیت می شود و در صورتی که تعداد درخواستها برای اتصال بیش از ظرفیت پاسخگویی باشد ،پراکسی سرور با کمبود منابع مواجه شده شده و دیگر نمی تواند به درخواستها پاسخ دهد و یک نفوذگر به راحتی می تواند در کار پراکسی سرور و در نتیجه شبکه توسط ایجاد ارتباطات زیاد اختلال شدید ایجاد کند بنابراین در صورتکیه دلیل موجهی برای اینکار ندارید از به کارگیری این دستور هنگام کامپایل خودداری کنید.

--disable-internal-dns

به منظور پاسخگویی سریعتر و بهتر به درخواستها اسکوئید از یک ابزار DNS اختصاصی و داخلی استفاده می کند و از آن برای سرعت بخشیدن به فرایند ترجمه نشانی های URL استفاده می کند که این امر موجب کاهش زمان تاخیر در پاسخها می شود.این قابیلت به طور پیش فرض فعال است و در صورتکیه بخواهید این ویژگی غیر فعال شود از این دستور استفاده کنید.

--enable-default-hostsfile

در بسیاری از توزیعهای لینوکس یک فایل متنی به نام hosts در مسیر /etc/ وجود دارد که نام ماشین یا میزبان را در آن نگهداری می کند. یکی از کاربرد های این نام این است که اسکوئید از آن در پیغام ها و صفحات خطایی که به کاربران نمایش می دهد استفاده می کند و به وسیله دستور ./configure – enable-default-hostsfile=/some/other/location/hosts می توان مسیر پیش فرض آن را تعیین کرد.

--enable-auth

اسکوئید از سیستم های احراز هویت مختلفی پشتیبانی می کندو این دستور موجب فعال شدن آن می شود البته این سیستم ها در حال تغییر بوده و احتمالا در نسخه های بعدی اسکوئید شاهد تغییراتی در آنها خواهیم بود . همچنین با استفاده از دستور ./configure –enable-auth=basic,digest,ntlm می توان نوع سیستم های احراز هویت مورد نظر را تعیین کرد البته این دستور از جمله دستورات قدیمی است که در نسخه های بالاتر اسکوئید می توانید از ./configure –enable-auth به جای دستور بالا استفاده کنید.

--enable-auth-basic

این دستور سیستم های پایه ای احراز هویت را فعال می کند که شامل مواردی چون LDAP، PAM، NCSA می باشد. برای فعال کردن این ویژگی از دستور ./configure –enable-auth-basic=PAM,NCSA,LDAP استفاده شود.توجه داشته باشید در صورتکیه نام ابزارهای مورد نظر خود را در این دستور وارد نکنید ،به طور پیش فرض تمامی ابزارهای موجود فعال می شوند.جهت آگاهی از فهرست ابزارهای موجود به پوشه helpers/basic_auth/ واقع در پوشه اصلی کد منبع اسکوئید مراجعه کنید.در صورتکیه تمایلی به فعال بودن این ابزارهای ندارید با استفاده از دستور --auth-basic-disable می توانید آن را غیر فعال کنید همچنین در صورتکیه می خواهید این ویژگی در حالت فعال باقی بماند اما هیچ کدام از ابزارها انتخاب نشوند دستور./configure --enable-auth-basic=none. این عمل را انجام می دهد.

--enable-auth-ntlm

ntlm ، هم یکی دیگر از سیستم های احراز هویت است که اسکوئید قادر به استفاده از آن است . با استفاده از دستور ./configure –enable-auth-ntlm=smb_lm,no_check می توان انواع مختلف ابزارهای مبتنی بر ntlm را انتخاب کرده و به همراه اسکوئید به کار گرفت همچنین جهت مشاهده ابزارهای موجود مبتنی بر ntlm به پوشه ی helpers/ntlm_auth/ واقع در پوشه ی کد منبع اسکوئید مراجعه کنید.

--enable-auth-negotiate

Negotiate یکی دیگر از ابزاهای احراز هویت است و نحوه عملکرد و توضیحات آن مانند موراد قبلی است و با استفاده از فرمان ./configure --enable-auth-negotiate=Kerberos. می توان آن را فعال کرد.

--enable-auth-digest

digest هم یکی دیگر از مکانیسم های احراز هویت است که توضیحات و نحوه عملکرد آن مشابه موارد قبل است.

--enable-ntlm-fail-open

در صورت استفاده از این قابلیت ، هنگامی که یکی از ابزارهای احراز هویت با مشکل مواجه شونداسکوئید همچنان به کار خود ادامه داده و کاربران را توسط سایر ابزارها احراز هویت می کند البته در استفاده از این ویژگی باید جوانب احتیاط را رعایت کرد زیرا موجب کاهش سطح امنیت شبکه می شود.

--enable-external-acl-helpers

یکی از ویژگیهای اسکوئید استفاده از Acl های خارجی است،این Acl ها را می توان داخل یک فایل متنی نوشته و مسیر آن را در فایل اصلی اسکوئید مشخص نمود.به منظور افزایش امنیت امکان استفاده از مکانیسم های احراز هویت نظیر ldap همراه با آنها پیش بینی شده است. فهرست ابزارهای ساخت Acl های خارجی در پوشه ی helpers/external_acl/ واقع در پوشه کد منبع اسکوئید قابل مشاهده است. برای فعال سازی این قابلیت می توان از دستور ./configure --enable-external-acl-helpers=unix_group,ldap_group استفاده کرد.

--disable-translation

همانطور که قبلا هم گفته شد اسکوئید در نمایش صفحات خطای خود از بسیاری از زبانهای زنده ی دنیا پشتیبانی می کند،به همین دلیل هنگامی که خطایی رخ می دهد تلاش می کند صفحه مورد نظر را به زبان محلی آن ناحیه ترجمه کند به طور مثال درصورتیکه پراکسی سرور شما در ایران قرار دارد اسکوئید تلاش می کند،صفحه خطای مورد نظر را به زبان پارسی ترجمه نماید درصورتیکه مایل به استفاده از این قابلیت نیستیدبه کارگیری این دستور موجب غیر فعال شدن این ویژگی می شود.

--disable-auto-locale

اسکوئید با استفاده از اطلاعات موجود در هدر های درخواست های دریافت شده ار کاربران سعی در یافتن زبان محلی منطقه ای که کاربران در آن قرار دارند می کند.این دستور موجب غیر فعال شدن این قابلیت می شود اما در صورتیکه می خواهید از این قابلیت استفده کنید باید برچسب error_directory که مسیر پوشه ی صفحات خطای اسکوئید را تعیین می کند به درستی تعیین شده باشد.

--with-default-user

هنگامی که اسکوئید راه اندازی می شود برای فعالیت خود در محیط سیستم عامل از یک نام کاربری استفاده می کند که این نام بسته به نوع سیستم عامل و روش نصب اسکوئید نامهای متفاوتی می تواند داشته باشد همچنین راه اندازی اسکوئید تحت مجوز کاربری ریشه که بالاترین سطح دسترسی در سیستم عاملهای لینوکس و یونیکس به لحاظ امنیتی کار صحیحی تلقی نمی شود به همین خاطر اسکوئید در حالت پیش فرض تحت مجوز نام کاربری nobody شروع به فعالیت می کند البته بسته به نوع سیستم عاملی که اسکوئید برروی آن فعال است این نام ممکن است proxy یا squid تعیین شده باشد. با استفاده از دستور configure –with-default-user=squid ./ می توان نام کاربری پیش فرض اسکوئید را از مقدار پیش فرض به مقدار دلخواه تغییر داد.

--with-logdir

سرویس دهنده اسکوئید به طور پیش فرض رویداد ها و خطاهای پیش آمده هنگام فعالیت را در فایلهای ثبت رخداد خود ذخیره می کند .مسیر نگهداری این فایلها معمولا به صورت ${prefix}/var/logs/تعیین شده است،درصورتیکه می خواهید

این مسیر پیش فرض را تغییر داده و مسیری دلخواه را برای آن انتخاب کنید می توانید از دستور -with -- configure./ logdir=/var/log/squid استفاده کنید.

--with-pidfile

مسیر پیش فرض ذخیره فایل pid اسکوئید در مسیر${prefix}/var/run/squid. pid قرار دارد این مسیر با مسیر نگهداری این فایل در بسیاری از توزیعهای لینوکس متفاوت است و در بسیاری از توزیعهای مشهور مسیر پیش فرض نگهداری این فایل در var/run/ قرار دارد. برای تغییر این مسیر دستور configure –with-pidfile=/var/run/squid.pid./

در نظر گرفته شده است.

--with-aufs-threads

درصورتکیه فرمت aufs را برای ساخت مخزن کش سرور انتخاب کرده اید می توانید از این دستور برای تعیین تعداد رشته ها استفاده کنید،درصورتکیه این ویژگی فعال نشود اسکوئید به طور خودکار تعداد رشته هایی را که باید مورد استفاده قرار بگیرد را محاسبه می کند . جهت فعال سازی این ویژگی از دستور configure –with-aufs-threads=12./ استفاده می شود.

--with-openssl

در صورتیکه می خواهید اسکوئید از این پروتکل پشتیبانی کند از دستور configure –with-openssl./استفاده کنید همچنین در صورتیکه openssl در مسیر پیش فرض خود نصب نشده است با استفاده از دستور -configure –with./ openssl=/opt/openssl می توان مسیر جدید را به صورت دقیق مشخص نمود.

--with-large-files

همانطور که پیش تر هم گفته شد اسکوئید از فایلهای ثبت رخداد برای نگهداری رخداد ها هنگام فعالیت استفاده می کند اما این فایلها تحت فشار شدید کاری و همچنین مدت فعالیت طولانی مدت به تدریج بزرگ شده و حجم آنها به چندین مگابایت خواهد رسید در حالیکه بسیاری از توزیع های لینوکس برای حجم فایلها محدودیت قائل می شوند و در صورتکیه حجم یک فایل بیش از یک مقدار معین افزایش یابد سیستم عامل با ایجاد محدودیت از افزایش حجم آن جلوگیری می کند درصورت بروز این مشکل فعالیت اسکوئید متوقف می شود. برای حل این مشکل می توان با به کارگیری ا دستور -with-large

files— محدودیت تعیین شده برای حجم فایلها توسط سیستم عامل را نادیده گرفت.

--with-filedescriptors

سیستم عامل ها از filedescriptor ها به منظور مسیر دهی و برقراری اتصالات و ارتباطات مورد نیاز نرم افزارهای مختلف استفاده می کنند.تعداد محدودی از filedescriptor ها در هر سیستم عامل برای یک حساب کاربری تعیین می شود (معمولا ۱۰۲۴)،اسکوئید از این فایلها برای برقراری ارتباطات و پاسخگویی به درخواستها استفاده می کند.در یک شبکه بزرگ و با ترافیک بالا این فایلها به سرعت مصرف می شوند و اسکوئید قادر به برقراری ارتباط بیشتر با کاربران و در نتیجه پاسخگویی بیشتر نیست و در این هنگام با بررسی فایل ثبت رویداد خطاهای اسکوئید در مسیر var/log/squid/cache.log/ پیغامی مبنی بر اتمام آن ها نمایش داده می شود، برای حل این مشکل از دستور ./configure –with-filedescriptors=8192. برای افزایش تعداد filedescriptor ها استفاده می شود که در اینجا این مقدار به ۸۱۹۲ افزایش داده شده است. برای بررسی تعداد موجود از دستور ulimit -n استفاده می شود که معمولا این مقدار به طور پیش فرض ۱۰۲۴ تعیین شده است. هنگامی که این مقدار را در اسکوئید مطابق دستور بالا افزایش می دهید طبیعتا باید این مقدار در سیستم عامل هم افزایش یابد، برای این منظور فایل /etc/security/limits.conf را به وسیله یکی از ویرایشگرهای متن مانند nano باز کرده و مقادیر ۸۱۹۲ username hard nofile و ۸۱۹۲ username soft nofile را به آن اضافه نمایید. توجه داشته باشید برای اعمال این تغییرات حتما باید از حساب کاربری ریشه استفاده کنید.

پیکربندی دستور configure/.

تاکنون گزینه های مختلفی را بررسی و به بیان خصوصیات هر یک پرداختیم برای به کارگیری هر کدام از قابلیتها و دستوراتی که در بخش قبل معرفی شدند باید هرکدام از آنها را به وسیله فرمانconfigure/. به اسکوئید معرفی شوند که در ادامه به انجام این کار می پردازیم:

```
./configure --prefix=/opt/squid/ --with-logdir=/var/log/squid/ --with-
pidfile=/var/run/squid.pid --enable-storeio=ufs,aufs --enable-removal-
policies=lru,heap --enable-icmp --enable-useragent-log --enable-referer-
log --enable-cache-digests --with-large-files
```

با اجرای دستور بالا در صورت فراهم بودن تمامی پیش نیازها ویژگی ها و گزینه های های انتخاب شده هنگام کامپایل به اسکوئید اضافه می شوند. در طول اجرای این پروسه در صورت بروز هر گونه مشکلی مانند نبود بعضی از کامپایلرها و کتابخانه ها ،دلیل بروز مشکل در خروجی نمایش داده می شود و عملیات کامپایل در همان مرحله متوقف می شود و پس از

برطرف کردن اشکالات به وجود آمده دوباره همان دستور را اجرا کرده تا عملیات کامپایل از سر گرفته شود.در صورتیکه همه چیز فراهم باشد پوشه های حاوی کدهای کامپایل شده اسکوئید (makefile) جهت ادامه مراحل بعدی نصب اسکوئید ساخته می شوند.توجه داشته باشید دستور بالا در واقع مجموعه ای قابلیتهایی را که هنگام کامپایل اسکوئید فعال و یا غیر فعال می شوند را تعیین می کند. تمامی گزارشهای اجرای این دستورات در فایلی به نام config.log در پوشه ی اصلی کامپایل ذخیره می شود و همواره جهت بررسی خطاهای به وجود آمده این فایل را بررسی کنید. در صورتکیه می خواهید دستورات دیگری را به مجموعه بالا اضافه کنید مانند الگوی بالا آنها را با یک خط فاصله به سایر گزینه های اضافه نمائید .در صورتیکه مرحله قبل با موفقیت به پایان برسد فایلهای اجرایی اسکوئید واقع در پوشه های مختلف جهت اجرای مرحله ی بعدی کامپایل در مسیرهای تعیین شده ذخیره می شوند. برای شروع مرحله دوم کامپایل کافیست دستور make all را در ترمینال وارد کنید برای اجرای این مرحله نیازی به مجوز ریشه نیست و این مرحله با توجه به منابع سخت افزاری موجود و همچنین ویژگیهای انتخاب شده حدود ۵ دقیقه طول می کشد در طول اجرای این دستور کدهای زیادی در صفحه نمایش که نشان دهنده کامپایل کدهای مختلف است نمایش داده می شود.درصورتیکه این مرحله نیز با موفقیت به اتمام برسد پیامی مشابه خطوط زیر برروی صفحه ی نمایش ظاهر می شود:

```
Making all in compat
make[1]: Entering directory '/home/user/squid-source/compat'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/home/user/squid-source/compat'
```

کدهای بالا نشان دهنده موفقیت آمیز بودن مرحله ی دوم کامپایل است.

مرحله سوم که در واقع آخرین مرحله نصب اسکوئید محسوب می شود فایلهای اجرایی ایجاد شده توسط دستور make all برروی سیستم عامل نصب می شود، برای انجام این مرحله ابتدا وارد حساب کاربری ریشه شده و با وارد کردن فرمان make install آخرین مرحله از پروسه کامپایل اسکوئید آغاز می شود در صورتکیه همه چیز به خوبی پیش برود پس از پایان این مرحله سرویس دهنده اسکوئید آماده ی پیکربندی و اجرا است.

بررسی فایلهای اسکوئید

اگر به پوشه نصب اسکوئید نگاهی بیندازید متوجه خواهید شد که این پوشه از چندین فایل متنی و زیر پوشه های مختلف تشکیل شده است برای مشاهده این پوشه ها می توان از دستور tree استفاده کرد که در این صورت تمامی فایلهای موجود در

پوشه ها به صورت شاخه های درخت نمایش داده می شود درصورتکیه بسته tree برروی سیستم عامل شما نصب نیست می توان از ابزار ls استفاده کرد.در ادامه به بررسی محتوای موجود در پوشه ی نصب اسکوئید می پردازیم.

bin

این پوشه شامل ابزارها و برنامه هایی است که بدون نیاز به مجوز ریشه قابلیت اجرا و راه اندازی تحت مجوز همه ی حساب های کاربری را دارا هستند.

bin/squidclient

squidclient یکی از برنامه هایی است که از آن برای بررسی صحت عملکرد اسکوئید استفاده می شود. برای اجرای این برنامه از دستور ${prefix}/bin/squidclient استفاده کنید که در اینجا با توجه به مسیر نصب اسکوئید مسیر دلخواه خود را به جای عبارت {prefix} وارد کنید.

etc

در این پوشه تمامی فایلهای پیکربندی اسکوئید نگهداری می شود که شامل موراد زیر است:

/etc/squid.conf

squid.conf فایل پیکربندی اصلی اسکوئید است که در طول مرحله نصب ایجاد شده و شامل تنظیمات پایه ای و پیشرفته جهت اجرای اسکوئید است. البته این تنظیمات معمولا دستخوش تغییرات زیادی می شوند و در طول مراحل پیکربندی اسکوئید گزینه های مختلفی به آن افزوده می شود.

/etc/squid.conf.default

این فایل متنی یک کپی از تمام تنظیمات پایه ای موجود در squid.conf را نگهداری می کند و در مواردی که به هر دلیلی تنظیمات اصلی از بین برود به عنوان پشتیبان برای راه اندازی دوباره نگهداری می شود.

/etc/squid.conf.documented

این فایل متنی برخلاف دو مورد قبل شامل هزاران خط توضیح و آموزش درمورد تنظیمات مختلف اسکوئید است و تمامی موارد موجود در فایل squid.conf را به صورت کامل و با بیان مثال شرح داده است همیشه پس از نصب اسکوئید برای مشاهده تغییرات صورت گرفته در نسخه ی جدید باید به این فایل مراجعه کنید به زبان ساده تر این فایل نحوه پیکربندی تنظیمات اسکوئید را همراه با ارائه ی مثال های مختلف از هر بخش توضیح می دهد.

Libexec

این پوشه شامل ابزارهای اجرایی اسکوئید است که در طول مراحل کامپایل ایجاد شده اند.

libexec/cachemgr.cgi

cache manager رابط کاربری گرافیکی است که به وسیله مرورگر اجرا شده و از آن می توان برای مشاهده وضعیت اسکوئید استفاده کرد .

sbin

این پوشه هم شامل نرم افزارهایی است که تنها به وسیله مجوز ریشه می توان آنها را اجرا کرد.

sbin/squid

/sbin/squid در واقع فایل اجرایی اصلی اسکوئید است که از آن برای شروع فعالیت ، راه اندازی مجدد و توقف فعالیت آن استفاده می شود.

share

در این پوشه فایلهای مختلفی از جمله مستندات ، صفحات خطای اسکوئید نگهداری می شود همچنین فایلهای مربوط به ایجاد تغییرات در رنگ و قالب این صفحات با فرمت css و html موجود است.

share/errors

در این پوشه تمامی صفحات خطای اسکوئید در قالب چندین زبان مختلف نگهداری می شود. شما می توانید با استفاده از یک ویرایشگر صفحات HTML موجود در آن را ویرایش کرده و توضیحات خود را به آن اضافه کرده و به نوعی آن را سفارشی سازی کنید.

share/icons

این پوشه حاوی آیکونهای کوچکی است که هنگام استفاده از پروتکل FTP در صفحه ایجاد ارتباط برای کاربر نمایش داده می شوند.

share/man

این پوشه محل نگهداری مستندات راهنمای اسکوئید و ابزارهای مربوط به آن است که در مراحل نصب اسکوئید ایجاد شده اند. با استفاده از دستورman در توزیع های مبتنی بر لینوکس و یونیکس به همراه نام و مسیر ابزار مورد نظرمیتوان به این مستندات دسترسی پیدا کرد. برای مثال برای مشاهده مستندات مربوط به اسکوئید با استفاده از دستور man /opt/squid/share/man/man8/squid.8 می توان محتوای آن را مشاهده کرد.همچنین جهت کسب اطلاعات

بیشتر د ر این زمینه به نشانی http://en.wikipedia.org/wiki/Man_page مراجعه فرمائید.

var

این پوشه محل قرارگیری فایلهایی است که در طول فعالیت اسکوئید به طور مداوم دچار تغییرات می شوند مانند دایرکتوری های مخزن کش و فایلهای ثبت رخداد که همگی در این پوشه نگهداری می شوند.

var/cache

این پوشه محل ذخیره سازی داده های درخواست شده توسط کاربران است که به طور پیش فرض ایجاد می شود و شما می توانید آن را تغییر دهید در واقع این پوشه محل ساخت پوشه های مخزن کش است.

var/logs

محل پیش فرض نگهداری فایلهای cache.log و access.log و سایر فایلهای ثبت رخداد است .

نصب اسکوئید از طریق بسته های آماده (باینری)

بسته های از پیش کامپایل شده ی اسکوئید در مخازن نرم افزاری بسیاری از توزیعهای لینوکس در دسترس بوده و از طریق ابزارهای مدیریت بسته های نرم افزاری توزیعهای مختلف قابل نصب است. در ادامه به بررسی چگونگی نصب و راه اندازی آنها در توزیعهای مختلف می پردازیم.

نصب اسکوئید در توزیعهای فدورا ، ردهت و سنت او اس

YUM سیستم مدیریت نرم افزاری توزیعهای مبتنی بر سیستم عامل ردهت مانند فدورا و CentOS است . بسته های مورداستفاده در این توزیعها با فرمت rpm نام گذاری می شوند. برای نصب اسکوئید در این توزیعها از دستور yum install squid استفاده کنید.

نصب اسکوئید در توزیعهای دبیان و اوبونتو

برای نصب اسکوئید در دبیان و اوبونتو از دستور apt-get به صورت apt-get install squid3 استفاده کنید. نکته ای که در اینجا باید به آن اشاره شود این است که در مخازن نرم افزاری توزیعهای دبیان و ابونتو و به طور کلی توزیعهای مبتنی بر دبیان، نسخه ی شماره ی ۳ اسکوئید با نام squid3 و نسخه ی ۲.۷ آن با نام squid نامگذاری می شوند.

نصب در توزیع FreeBSD

برای نصب اسکوئید از طریق توزیع FreeBSD می توان از دستور pkg_add -r squid31 استفاده کرد.جهت کسب

اطلاعات بیشتر در مورد پورت های مختلف نرم افزاری این توزیع به آدرس-www.freebsd.org/doc/handbook/packages using.html مراجعه نمائید.

نصب در توزیعهایNetBSD و OpenBSD

نصب اسکوئید از طریق توزیعهای NetBSD،OpenBSD مشابه توزیع FreeBSD است و با دستور pkd_add squid31 نصب می شود همچنین برای کسب اطلاعات و توضیحات بیشتر در این زمینه به نشانی www.openbsd.org/ports.html#Getو www.netbsd.org/docs/pkgsrc/using.html#installing-binary-packages مراجعه نمائید.

نصب در توزیعDragonfly BSD

برای نصب اسکوئید در این توزیع از دستور pkg_radd squid31 استفاده کنید.

نصب در توزیع جنتو

برای نصب اسکوئید در توزیع جنتو از دستور emerge =squid-3.1* استفاده می شود.

نصب در توزیع آرچ لینوکس

برای نصب اسکوئید در توزیع آرچ از نرم افزار مدیریت بسته های نرم افزاری pacman استفاده می شود و از طریق دستور pacman -S squid می توانید آن را نصب نمائید.همچنین جهت کسب اطلاعات و توضیحات بیشتر در این مورد به نشانی https://wiki.archlinux.org/index.php/Pacman. مراجعه نمائید.

خلاصه فصل

در این فصل مطالبی مختلفی در مورد پراکسی سرور ها ، نحوه عملکرد و دلایل به کارگیری آنها در شبکه عنوان شد. پراکسی سرور ها از طریق صرفه جویی در مصرف پهنای باند،افزایش سرعت بارگذاری صفحات وب موجب افزایش کیفیت بازدید کاربران از وب سایتها و تبادل اطلاعات میان آنها و سرورهای مقصد می شود. همچنین نحوه نصب و کامپایل پراکسی سرور اسکوئید از طریق کد منبع و بسته های موجود در توزیعهای مختلف لینوکس و یونیکس به طور کامل مطرح شد. به طور کلی مطالب مطرح شده در این فصل شامل موارد زیر است:

- آشنایی با مفهوم پراکسی سرور
- معرفی پراکسی سرور اسکوئید
- شرح مراحل و تنظیمات نصب اسکوئید
- کامپایل اسکوئید از طریق کد منبع
- نصب اسکوئید از طریق کد منبع و بسته های آماده
- شرح کامل کامپایل اسکوئید از طریق کد منبع
- معرفی و شرح مهمترین گزینه های موجود هنگام نصب اسکوئید از طریق کد منبع

فصل دوم

پیکربندی اسکوئید

اکنون که چگونگی نصب اسکوئید را آموختیم در ادامه می خواهیم به بررسی چگونگی پیکربندی آن مطابق نیازهای خود در شبکه بپردازیم. در این فصل به طور مفصل به بررسی فایل پیکربندی اسکوئید و همچنین گزینه های مختلف پیکربندی و چگونگی راه اندازی و بهینه سازی آن می پردازیم.

به طور کلی در این فصل در مورد موضوعات زیر بحث خواهد شد:

- ✓ راه اندازی سریع اسکوئید
- ✓ ساختار و گزینه های مختلف پیکربندی
- ✓ آشنایی با درگاه HTTP و نکات مهم در مورد پیکربندی
- ✓ بررسی نحوه عملکرد Acl ها
- ✓ کنترل دسترسی اسکوئید بر محتوا
- ✓ بررسی نحوه ارتباط با کش سرور های همسایه و همکار
- ✓ نحوه ذخیره سازی اطلاعات برروی حافظه موقت و دیسک سخت
- ✓ بهینه سازی اسکوئید به منظور کاهش مصرف پهنای باند و زمان تاخیر
- ✓ بررسی چگونگی دریافت و ارسال اطلاعات
- ✓ پیکربندی اسکوئید جهت استفاده از DNS سرور ها
- ✓ مطالب مختصری در مورد فایلهای ثبت رخداد

راه اندازی سریع پراکسی سرور

پیش از آنکه به بررسی جزئیات پیکربندی بپردازیم ابتدا نگاهی به حداقل دستورات مورد نیاز پیکربندی برای راه اندازی سریع پراکسی سرور می اندازیم. برای راه اندازی اسکوئید با حداقل تنظیمات مورد نیاز ابتدا فایل squid.conf را در مسیر /etc/squid/ و در مسیر اصلی نصب به وسیله یک ویرایشگر متن باز کرده و خطوط زیر را به آن اضافه کنید:

```
cache_dir ufs /usr/local/squid/var/cache 500 16 256
acl my_machine src 192.0.2.21 # Replace with your IP address
http_access allow my_machine
```

برای این کار خطوط مورد نظر را به ابتدای فایل پیکربندی اضافه کنید،توجه داشته باشید نشانی IP مورد نظر خود را به جای IP تعیین شده مثال بالا وارد کنیدودرمرحله بعد باید پوشه های مخزن کش ساخته شوند،با دستور/usr/local/squid/sbin/squid -z دایرکتوری های موردنظر ساخته می شوند.اکنون پراکسی سرور با وارد کردن دستور /usr/local/squid/sbin/squid/ در خط فرمان سیستم عامل راه اندازی خواهد شد. با آغاز به کار اسکوئید ، پراکسی سرور در شبکه فعال شده و با استفاده از درگاه پیش فرض تعیین شده خود یعنی ۳۱۲۸ می تواند به تمامی درخواستهای ماشینهای موجود در شبکه گوش دهد. برای استفاده از پراکسی سرور کافی است نشانی ip پراکسی سرور خود را همراه با شماره درگاه ۳۱۲۸ در قسمت تنظیمات پراکسی مرورگر خود وارد کنید. در صورتکیه نشانی http://www.example.com:897 را در مرورگر خود وارد نمائید پیغامی مبنی بر عدم دسترسی به این نشانی از سوی پراکسی سرور نمایان می شود که این خطا بیانگر آن است شما درحال استفاده از پراکسی سرور اسکوئید هستید.حال در ادامه می خواهیم با جزئیات گزینه های موجود در فایل پیکربندی اسکوئید بیشتر آشنا شویم.

ساختار فایل پیکربندی

فایل پیکربندی اسکوئید معمولا در یکی از مسیرهای /usr/local/squid/etc/squid.conf،etc/squid/squid.conf/ ویا $ {prefix}/etc/squid.conf که در اینجا منظور از prefix مسیری است که در هنگام کامپایل با استفاده از دستور prefix-- تعیین شده است (فصل گذشته را به خاطر بیاورید) مسیرهای از پیش تعریف شده به طور پیش فرض هنگامی که شما یک مسیر خاص را هنگام کامپایل اسکوئید مشخص نکرده باشید تعیین می شوند که اولی هنگام کامپایل و دومی زمان استفاده از بسته های آماده است.در نسخه های جدید اسکوئید فایل پیکربندی دیگری به نام squid.conf.documented وجود دارد

که در آن به تشریح دستورات و گزینه های مطرح در پیکربندی اسکوئید می پردازد و درآن تمامی ویژگی ها و دستورات موجود برای پیکربندی اسکوئید به همراه مثال و توضیحات شرح داده شده است،همچنین همواره جهت آگاهی از تغییرات به وجود آماده در دستورات پیکربندی و قابلیت های جدید مطالعه محتوای این فایل ضروری است. در این فصل بخش از تنظیمات و دستورات پیکربندی اسکوئید شرح داده می شوند ، جهت کسب اطلاعات بیشتر در این زمینه به آدرس http://www.squid-cache.org/Doc/config مراجعه نمائید.از نظر ساختاری این فایل راهنما مانند سایر راهنماهای موجود در سایر نرم افزارهای یونیکس و لینوکس است به علاوه مثالهایی برای درک بهتر موضوعات بیان شده است و ساختار آن به گونه ای است که خطوطی که قبل از آنها # آمده است به عنوان توضیح در نظر گرفته می شود و هنگام راه اندازی اسکوئید به طور کامل نادیده گرفته خواهند شد. . به مثال زیر توجه کنید:

```
# TAG: cache_effective_user
# If you start Squid as root, it will change its effective/real
# UID/GID to the user specified below. The default is to change
# to UID of nobody.
# see also; cache_effective_group
#Default:
# cache_effective_user nobody
```

مثال بالا که یک قطعه از دستورات موجود در فایل squid.conf.documented است نحوه بیان دستورات را نشان می دهد به طوریکه در خط اول و در قسمت TAG نام دستور مورد نظر که در اینجا cache_effective_user است مشخص می شود در خطوط بعدی به معرفی و توضیح خصوصیات دستور می پردازد و در قسمت Default مقدار پیش فرضی که در تنظیمات اسکوئید برای آن تعیین شده است را مشخص می کند .همانطور که مشاهده می کنید با مطالعه این توضیحات به راحتی می توان با نحوه کاربرد و عملکرد دستورات مختلف آشنا شد.

ساختار دستورات

دستورات مورد استفاده در فایل پیکربندی بسته به نوع و نحوه کاربردشان از ساختار خاصی پیروی می کنند ،برخی از آنها تک بخشی وچند بخشی هستند و برخی دیگر فقط مقادیر منطقی و صحیح را به عنوان ورودی می پذیرند.

دستوراتی که فقط یک ورودی را می پذیرند

از این دستورات می بایست تنها یک بار در طول فایل پیکربندی استفاده کرد و در صورت استفاده ی بیش از یک بار ، تمامی مقادیر قبلی نیز به مقدار جدید تغییر می بایند و تنها مقدار جدید اعمال می شوند.logfile_rotate یکی از این دستورات است که فقط یک بار باید مقدار دهی شود.در مثال زیر نحوه مقدار دهی به دستور logfile_rotate را نشان می دهد:

```
logfile_rotate 10
# Few lines containing other configuration directives
logfile_rotate 5
```

در مثال بالا این دستور دو بار مقدار دهی شده است که در نهایت مقدار logfile_rotate 5 به عنوان مقدار نهایی اعمال می شود، زیرا اسکوئید فایل پیکربندی خود را از بالا به پایین میخواند.

دستورات با مقادیر ورودی منطقی

این دستورات هم مانند نوع قبل فقط یک بار مقداردهی می شوند با این تفاوت که در این نوع فقط مجاز به استفاده از گزینه های on و off هستیم یعنی این دستورات تنها می توانند یکی از مقادیر on یا off را بپذیرند و بقیه مقادیر بی معنی تلقی می شوند.

```
query_icmp on
log_icp_queries off
url_rewrite_bypass off
```

کاربرد این دستورات معمولا زمانی است که میخواهیم یک مقدار پیش فرض تعیین شده را تغییر دهیم.

دستوراتی که بیش از یک مقدار را می پذیرند

دستورات در نظر گرفته شده برای این نوع می توانند بیش از یک مقدار داشته باشند.در این نوع می توان تمامی مقادیر را در یک خط و بصورت پشت سر هم و با یک فاصله قرار داد و یا می توان هر یک را در یک خط جدید و با فواصل مختلف تعریف کرد.

```
hostname_aliases proxy.exmaple.com squid.example.com
```

می توانیم مقادیر موجود در خط بالا را از هم تفکیک کرده و در دو خط جداگانه تعریف کنیم:

```
dns_nameservers proxy.example.com
```

```
dns_nameservers squid.example.com
```

در مثال بالا همانطور که مشاهده می کنید ،مقادیر مورد استفاده برای هر کدام از دستورات می توانند به صورت خطی و پشت سر هم در یک دستور و یا در خطوط مجزا تعریف شوند در واقع در مثال قبل اسکوئید مقادیر موجود در خط اول را با مقادیر موجود در خطوط دوم و سوم یکسان فرض می کند.

دستورات زمانی

در این دستورات واحد قابل قبول برای مقدار دهی زمان است.اسکوئید واحد های زمانی ساعت،دقیقه و ثانیه را تشخیص می دهد و هنگام استفاده باید نوع واحد مورد نظر نیز ذکر شود .به مثال زیر توجه کنید:

```
request_timeout 3 hours
persistent_request_timeout 2 minutes
```

در خط اول واحد زمانی ساعت و خط در خط دوم واحد زمانی دقیقه در نظر گرفته شده اند.

دستورات با مقادیر ورودی اندازه فایل یا حافظه

هنگام پیکربندی اسکوئید لازم است مقدار حافظه اصلی و دیسک سخت را تعیین کرد بدین منظور می توان از این دستورات که به طور مستقیم برای تعیین محدوده اختصاص منابع سیستم به کار می رونداستفاده کرد. برای تعیین مقادیر می توان از واحد های بایت ،کیلوبایت،مگابایت و یا گیگا بایت استفاده کرد:

```
reply_body_max_size 10 MB
cache_mem 512 MB
maximum_object_in_memory 8192 KB
```

همانطور که می بینید برای تعیین مقادیر مورد نظر از واحد های مگابایت و کیلو بایت استفاده شده است که هر دو قابل قبول هستند.اکنون که با ساختار پیکربندی آشنا شده ایم فایل پیکربندی اصلی اسکوئید یعنی squid.conf را باز کرده و به بررسی سایر گزینه های پیکربندی می پردازیم.

درگاه HTTP

در این دستور شماره درگاهی که اسکوئید از آن برای گوش کردن به درخواستها و اتصالات کاربران استفاده می کند تعیین می شود که مقدار پیش فرض تعیین شده برای آن ۳۱۲۸ است .راههای مختلفی برای تعیین شماره درگاه HTTP وجود دارد که

عبارتند از:

- در ساده ترین حالت کافی است شماره درگاه مورد نظر خود را مقابل عبارت http_port واردکنید مثلا :

http_port 8080

- در صورتیکه برروی پراکسی سرور بیش از یک کارت شبکه وجود دارد و میخواهیم فقط کاربران شبکه LAN از پراکسی سرور استفاده کنند می توان با استفاده از دستور http_port 192.0.2.25:3128 آدرس ip کارت شبکه مورد نظر را به همراه شماره درگاه تعیین کرد. در این حالت فقط کارت شبکه ای که این آدرس را گرفته می تواند به اتصالات و درخواست های کاربران گوش کند.

- یکی دیگر از راهها ،استفاده از نام میزبان (host name) به جای آدرس ip مانند مثال زیر:

http_port myproxy.example.com:8080 در این حالت نام میزبان به نشانی ip مورد نظر ترجمه می شود.

- در صورتکیه می خواهیم پراکسی سرور بیش از یک آدرس و درگاه برای گوش کردن به اتصالات شبکه داشته باشد،می توان هرکدام از آدرسها را همراه با درگاه تعیین شده در چند سطر زیر هم قرار داد بدین ترتیب اسکوئید از طریق هر یک از آدرس ها می تواند به درخواستهای کاربران با آدرسهای مختلف گوش کند.این روش زمانی کاربرد دارد که در شبکه تعدادی گروه کاربر LAN با آدرس های مختلف وجودداشته باشد و بخواهیم هرکدام از این گروهها از یک آدرس و درگاه برای دسترسی به شبکه استفاده کنند مانند مثال زیر:

http_port 192.0.2.25:8080

http_port 10.10.250.21:3128

http_port lan2.example.com:8081

در این مثال اسکوئید می تواند از طریق سه آدرس و سه درگاه مختلف به درخواستهای کاربران گوش کند.

- در نسخه های به روز شده ی اسکوئید گزینه های جدیدی مانند intercept, tproxy, accel اضافه شده اند ، در حالت Intercept پراکسی سرور می تواند درخواستهای کاربران را تفکیک کند و به صورت شفاف میان کاربران و شبکه ی خارجی قرار گیرد بدون اینکه نیازی به پیکربندی مجدد کاربران و تنظیمات پراکسی باشد در این مورد در فصل ۱۰ به طور مفصل بحث خواهیم کرد برای استفاده از این گرینه از دستور http_port 3128 intercept استفاده می شود.

Tproxy یکی دیگر از قابلیت های جدید است که می تواند آدرس های ip کاربران هنگام عبور از پراکسی سرور را بدون

تغییر عبور دهد برای این کار از دستور http_port 8080 tproxy استفاده می شود. در حالت شتاب دهنده یا Accelerator mode از دستور http_port 80 accel defaultsite=website.example.com استفاده می شود در این مورد در فصل ۹ مطالب بیشتری را خواهیم آموخت.

فهرست های کنترل دسترسی

Access Control Lists یا ACL ها از جمله پرکاربردترین دستورات کنترلی هستند که از آنها برای کنترل دسترسی کاربران به شبکه ،کنترل دسترسی به سایتها و ...استفاده می شود. این دستورات به توجه به انعطافی که دارند همواره به صورت ترکیبی همراه با http_access و icp_access ها به کار می روند. هر Acl تعریف شده باید دارای یک نام و نوع باشد،انتخاب نام معمولا اختیاری بوده و نوع آن هم با توجه به نوع کاربرد مورد نظر ما انتخاب می شود و عموما در انتهای تمامی ACL ها برای اعمال محدودیت ها از http_access deny|allow استفاده می شود که در واقع این قسمت به عنوان فعال کننده ACL شناخته می شود.

به طور کلی برای تعریف یک ACL از قاعده کلی زیر استفاده می شود:

```
acl ACL_NAME ACL_TYPE value
acl ACL_NAME ACL_TYPE "/path/to/filename"
```

در اینجا به جای ACL_NAME نام ACL مورد نظر ،و به جای ACL_TYPE نوع ACL مشخص می شود و در قسمت value نیز مقدار ورودی قرار می گیرد که این مقدار می تواند یک یا چندین فرمان قابل اجرا باشد و همانطور که در خط دوم مشاهده می کنید می توان مقادیر را در یک فایل متنی ذخیره کرده و مسیر فایل را به جای مقدار قرار داد. استفاده از روش دوم برای ساخت دستورات و عبارات طولانی بسیار مفید است.

شیوه ساخت ACL های ساده

همانطور که گفته شد،ACL ها دارای انواع مختلفی هستند،درصورتیکه بخواهیم عملیاتی را برروی یک وب سایت انجام دهیم مثلا دسترسی به آن را محدود کنیم و یا آن را به نشانی دیگری ارجاع دهیم از ACL نوع dstdomain استفاده می کنیم،در این نوع مقدار مورد نظر باید حتما از نوع domain باشد .به طور مثال میخواهیم دسترسی به وب سایت example.com را محدود کنیم برای این کار به صورت زیر عمل می کنیم:

```
acl example dstdomain example.com
http_access deny example
```

در این مثال نام Acl تعریف شده example و از نوع dstdomain است و مقدار تعیین شده برای آن example.com است،در خط دوم نوع عملیاتی که قرار است انجام شود تعیین می شود که در اینجا ایجاد محدودیت در دسترسی کاربران به این آدرس است. تقریبا در تمامی Acl هایی که به منظور ایجاد کنترل و محدودیت در یک زمینه ی خاص مورد استفاده قرار می گیرند از برچسب http_access به منظور اعمال کنترل و محدودیت استفاده می شود.

برای ساخت یکAcl سه راه کلی به شرح زیر وجود دارد :

۱. کلیه ی مقادیر مورد نظر را در یک خط دستور قرار دهیم : acl example_sites dstdomain example.com example.netدر اینجا Acl مورد نظر ما دو مقدار گرفته است و هر دو با یک فاصله پشت سر هم قرار گرفته اند. استفاده از این نوع زمانی که تعداد ورودی های ما محدود هستند مناسب است.

۲. مانند سایر دستوراتی که معرفی کردیم می توان از یکAcl در چند خط مجزا و با یک نام یکسان استفاده کرد:

```
acl example_sites dstdomain example.com example.net
acl example_sites dstdomain example.org
```

۳. مقادیر در یک فایل متنی: در صورتیکه تعداد مقادیری که قرار است در یک Acl قرار داده شود خیلی زیاد باشد می توان از یک فایل متنی جهت قرار دادن دستورات در آن استفاده کرد و در انتها کافی است در قسمت وارد کردن مقادیر متغیر ها مسیر فایل متنی ای که دستورات در آن قرار گرفته اند را وارد کرد مانند مثال زیر:

```
acl example_sites dstdomain '/opt/squid/etc/example_sites.txt'
```

برای ساده تر شدن کار و جلوگیری از گم کردن مسیر ساخت فایلها بهتر است فایلهای متنی خود را در همان پوشه etc/squid/ قرار داد در این صورت دسترسی به آن آسان تر خواهد بود.برای مثال محتوای دستورات موجود در فایل example_sites.txt مانند خطوط زیر است:

```
# This file can also have comments
# Write one value (domain name) per line
example.net
example.org # Temporarily remove example.org from example_sites
aclexample.com
```

Acl ها نسبت به کوچک و بزرگ بودن حروف نامگذاری حساس نیستند بنابراین با رعایت نوع Acl انتخابی می توان از یک نوع Acl با نامهای متفاوت در فایل پیکربندی اسکوئید استفاده کرد:

```
acl NiCe_NaMe src 192.0.2.21
acl nIcE_nAmE src 192.0.2.23
```

در نگاه اول ممکن است این چنین به نظر آید که این دو،acl های متفاوتی هستند در حالیکه چنین نیست در واقع هر دو یک acl محسوب می شوند و کوچک و بزرگ بودن حروف تاثیری در نامگذاری آنها نخواهد داشت و هردو با نام nice شناخته می شوند.مثال هایی که در بالا آورده شد،ساده ترین نوع acl ها بودند و ساختار کاملا پایه ای محسوب می شوند.موارد پیشرفته تر را در فصل چهار کتاب بررسی خواهیم کرد.

نکته: یکی از مهمترین نکاتی که در نامگذاری **Acl** *ها باید رعایت شوداین است که از یک نام منحصر به فرد نباید در بیش از یک نوع* **Acl** *متفاوت استفاده شود. مثلا:*

```
acl destination dstdomain example.com
acl destination dst 192.0.2.24
```

این دستورات اشتباه هستند زیرا نوع آنها با هم متفاوت است و در صورت به کارگیری چنین مواردی هنگام راه اندازی موجب بروز خطا و اشکالات غیر قابل پیش بینی دیگری می شود.

مدیریت دسترسی به پراکسی سرور

هنگامی که اسکوئید برروی سرور راه اندازی شود راههای مختلفی برای دسترسی به آن وجود دارد مانند استفاده از یک مرورگر معمولی،استفاده از آن به عنوان پراکسی سرور همسایه و یا به وسیله سرور های دیگر. سرویس دهنده اسکوئید برای کنترل سطوح دسترسی راههای مختلفی را ارائه کرده است که در ادامه به بررسی راههای جلوگیری و یا صدور مجوز دسترسی به سرور پراکسی می پردازیم.

کنترل دسترسی به پروتکل HTTP

Aclها به تنهایی قادر به انجام هیچ گونه کنترلی بر محتوای موجود در شبکه نیستند تنها کاری که آنها انجام می دهند شناسایی و نشانه گذاری آدرسها و مسیر ها جهت به کارگیری اعمال تاثیر توسط سایر دستورها است به طور مثال با تعریف یک محدوده

ی ip توسط Acl به تنهایی نمی توان برروی آن کنترلی داشت .به بیان واضح تر Acl ها تنها به عنوان دستوراتی برای تعیین مواردی که میخواهیم بر آن ها کنترل داشته باشیم به کار می روند. برای اعمال کنترل و بعضا محدودیت باید از دستورات همراه با پسوند _access استفاده کرد که http_access یکی از پرکاربردترین نوع این دستورات است که از آن برای تعیین وضعیت دسترسی به همراه یک Acl به کار می رود.ساختار کلی http_access به صورت زیر است:

http_access allow|deny [!]ACL_NAME

برچسب http_access می تواند تبادل داده از طریق پروتکل http را کنترل کند.در قسمت ACL_NAME ، باید نام Acl ی قبلا تعریف شده است قرار گیرد همچنین در صورتی که قبل از نام Acl عبارت (!) قرار گیرد، درخواست های سایر کاربران و داده هایی که به نوعی با این Acl مرتبط نیستند بسته به نوع دسترسی ای که دستور http_access تعیین می کند جزء موارد استثنا محسوب می شوند مثلا اگر در اینجا نوع دسترسی deny یعنی عدم اجازه ی عبور ترافیک تعیین شده باشد در این صورت سایر کاربرانی که محدوده آدرس آنها در این Acl قرار نداشته باشد اجازه عبور ترافیک و بازدید از وب سایت ها را پیدا خواهند کرد.برای اینکه بتوانیم از دسترسی برخی کاربران به پراکسی سرور جلوگیری کنیم بایدآدرس ip کاربر مورد نظر در یک Acl همراه با http_access قرار دهیم این محدودیت ها می تواند علیه یک کاربر یا گروهی از کاربران در یک محدوده ی آدرس باشد این موضوع کاملا بستگی به وضعیت شبکه شما دارد. برای روشن شدن این موضوع به مثالهای زیر توجه کنید:

acl my_home_machine src 192.0.2.21

acl my_lab_machine src 198.51.100.86

http_access allow my_home_machine

http_access allow my_lab_machine

در اینجا در هردو Acl به دو آدرس ip مجوز دسترسی داده شده است برعکس ،اگر بخواهیم دسترسی این دو آدرس را محدود کنیم از برچسب deny استفاده می کنیم برای جلوگیری از ازدیاد خطوط فایل پیکربندی و دسترسی سریع به دستورات،می توان Acl هایی که برای مقاصد مشترک استفاده می شود را در هم ادغام کرد مثلا در مثال بالا از یک Aclبرای هر دو ip استفاده می کنیم :

acl my_machines src 192.0.2.21 198.51.100.86

http_access allow my_machines

این acl همان کاری که دو خط بالا انجام می دهند را به تنهایی انجام می دهد و عملا از نظر کاربرد هیچ تفاوتی با هم ندارند

فقط نوع دوم ساده تر و کاربردی تر است. برای جلوگیری از دسترسی کاربران ناشناس به سرور و جلوگیری از سوء استفاده از آن همیشه محدوده ی ip ای که در شبکه استفاده می شود را مشخص کرده و به وسیله یک Acl آن را در فایل پیکربندی تعریف کنید و دسترسی سایر کاربران را به پراکسی سرور خود محدود کنید این کار باعث می شود که سطح امنیتی شبکه شما به طور محسوسی افزایش یابد و همچنین از اتلاف پهنای باند توسط سایر کاربران جلوگیری می شود. مثال زیر این موضوع را بهتر نشان می دهد:

```
acl lan src 10.240.10.0/24
http_access allow lan
http_access deny all
```

در اینجا فرض بر این است محدوده ی ip کاربران در Acl ای به نام lan تعریف شده است ومیخواهیم غیر از این کاربران هیچ کاربر دیگری نتواند از پراکسی سرور استفاده کند در خط دوم اجازه دسترسی به این آدرس ها داده شده است و در خط سوم تمامی دسترسی ها را به جز محدوده ی ip موجود در Acl ی به نام lan مسدود شده است .در بسیاری از Acl های پیش فرض و از پیش تعریف شده که در فایل پیکربندی مشاهده می کنید عبارت allow all یا deny all دیده می شود ،عبارت all به معنای اعمال یک کنترل یا محدودیت بر تمامی آدرس ها و در نتیجه تمامی کاربران شبکه است به عبارت دیگر این دستورات تمامی ترافیک ورودی یا خروجی را تحت تاثیر قرار می دهند و کل شبکه را تحت تاثیر قرار می دهند.همچنین در حالت پیش فرض اسکوئید تمامی ترافیک شبکه را از طریق عبارت http_access deny all مسدود می کند این کار به دلایل امنیتی صورت می گیرد و شما ابتدا می بایست مقدار این دستور را مطابق نیازهای خود اصلاح کنید.

دسترسی به پاسخ های HTTP

پاسخ های HTTP در واقع پاسخ هایی هستند که از طرف یک وب سرور که با کاربران تبادل اطلاعات داشته است به سوی پراکسی سرور ارسال می شود. با استفاده از دستور http_reply_access می توان دسترسی به این پاسخ های دریافت شده را کنترل کرد.نحوه استفاده از این دستور کاملا شبیه http_access است و به صورت http_reply_access allow| deny [!]ACL_NAME به کار می رود .این دستور می تواند دسترسی هایی که از طریق http_access اعطا شده است را لغو نماید. به مثال زیر توجه کنید:

```
acl my_machine src 192.0.2.21
http_access allow my_machine
http_reply_access deny my_machine
```

در اینجا مطابق دسترسی اعطا شده توسط http_access کاربر می تواند به شبکه دسترسی داشته باشد اما در خط سوم در قسمت reply_access کاربر هنوز قادر به دریافت هیچ گونه پاسخی از جمله پاسخ درخواست خود نیست در واقع این کاربر نمی تواند محتوای وب سایتها را مشاهده کند دلیل آن هم این است که او اجازه ارسال درخواست به سوی پراکسی سرور را دارد اما اجازه ی دریافت پاسخ از او سلب شده است .از این دستور برای جلوگیری از دسترسی کاربران به محتوای چند رسانه ای موجود در وب سایتهای مختلف مانند فیلم ها،صداو ... استفاده می شود. همواره هنگام به کارگیری فرمان http_reply_access نهایت دقت را داشته باشید زیرا استفاده ی نادرست از این دستور موجب کاهش شدید امنیت شبکه و به خطر افتادن اطلاعات شخصی کاربران می شود دلیل این موضوع این است که مثلا فرض کنید یک کاربر در حال ارسال اطلاعات شخصی خود به سوی یک وب سایت از طریق HTTP POST است او اطلاعات خود را به سوی سرور ارسال می کند اما به دلیل وجود دستور http_reply_access deny قادر به دریافت هیچ گونه اطلاعاتی از سوی سرور نیست احتمالا تا اینجا فکر می کنید که این موضوع اشکالی ندارد اما حال تصور کنید وب سایتی که کاربر در حال ارسال اطلاعات حساس خود مانند کارت اعتباری است یک وب سایت است که توسط یک هکر کنترل می شود در این صورت به دلیل وجود ارتباط یک طرفه میان کاربر و سرور ،یک هکر باهوش به راحتی قادر به سرقت تمامی اطلاعات کاربر خواهد بود بدون اینکه کاربر مورد نظر کوچکترین خطری را احساس نماید.

دسترسی به پروتکل ICP

این پروتکل به طور اختصاصی به مظنور برقراری ارتباط اسکوئید با سایر کش سرورهای همسایه استفاده می شود. Internet cache protocol یا (ICP) به منظور ارتباط موثر کش سرورها با یکدیگر طراحی شده است و معمولا برای تعیین شماره درگاه ICP به کار می رود.ساختار نحوی این دستور نیز شبیه http_access بوده و به صورت زیر است :

```
icp_access allow|deny [!]ACL_NAME
```

همچنین مقدار پیش فرض آن deny یعنی مسدود کردن تمامی ارتباطات با کش سرور های همسایه از طریق این درگاه و پروتکل است .

دسترسی به پروتکل HTCP

ساختار این دستور هم مشابه موارد قبلی است و از آن برای پاسخ و یا عدم پاسخ به درخواستهای HTCP ارسال شده از سوی سرورهای همسایه به کار می رود و مقدار پیش فرض آن نیز deny است.

دسترسی به HTCP CLR

کش سرور های همسایه از طریق ارسال پیامهای HTCP CLR به سایر همسایگان خود می توانند برخی از داده های موجود در مخزن کش سرور را حذف و یا پالایش کنند که این داده ها ممکن است از نوع داده هایی باشد که مدت زمان زیادی از تاریخ ذخیره آن ها گذشته است و یا دیگر منابع آنها موجود نیست .با استفاده از دستور htcp_clr_access می توان این امکان را برای سرورهای قابل اطمینان فعال کرد.

Miss access

از این دستور برای تعیین کش سرورهای همسایه ای که می توانند درخواستهای کاربران خود را از طریق کش سرور شما پاسخ دهند به کار می رود. هنگامی که درخواستی از سوی یک به وسیله کش سرور دریافت می شود ،در صورتیکه داده درخواستی قبلا هم توسط کاربر دیگری درخواست شده باشد و برچسب HIT را به همراه داشته باشد کش سرور برای پاسخ به آن از داده های موجود در مخزن خود استفاده می کند و در صورتیکه شیء مورد نظر در مخزن کش موجود نباشد و یا به عبارتی داده جدید باشد برچسب Miss را دریافت کرده و جزء داده های کش نشده محسوب می شود و کش سرور برای پاسخگویی به آن باید با سرور مقصد ارتباط برقرار نماید. هنگامی که در یک شبکه چند کش سرور وجود داشته و با هم در ارتباط باشند از داده های ذخیره شده در مخزن خود به منظور پاسخگویی به درخواستهای کاربران سرورهای همسایه که درخواستهای کاربران را به سوی سایر سرورها ارسال کرده اند استفاده می کنند در صورتکیه بخواهیم به برخی از سرورهای همسایه اجازه ی ارسال درخواستهای Miss را ندهیم از دستور زیر استفاده می کنیم:

```
acl bad_clients src 192.0.2.0/24
miss_access deny bad_clients
miss_access allow all
```

در اینجا به کاربرانی که در محدوده ی آدرس تعیین شده در bad_clients هستند اجازه استفاده از منابع پراکسی سرور شما نخواهند داشت.

Ident lookup access

این دستور تعیین می کند که اسکوئید به جستجو و شناسایی نام های کاربری تعریف شده برای کاربران هنگام پاسخگویی به

یک درخواست که از طریق پروتکل TCP ارسال می شود بپردازد و یا خیر که مقدار پیش گزیده ی آن عدم مراجعه برای جستجوتعیین شده است :

```
acl ident_aware_hosts src 192.0.2.0/24
ident_lookup_access allow ident_aware_hosts
ident_lookup_access deny all
```

در این مثال اسکوئید فقط به جستجوی نام های کاربری کاربران موجود در محدوده آدرس تعیین شده در ident_aware_hosts می پردازد.

نکته: فقط **Acl** *هایی که برپایه پروتکل* **TCP/IP** *هستند در این نوع* **Acl** *پشتیبانی می شوند.*

کش سرورهای همسایه و همکار

کش سرورهای همکار و یا همسایه به سایر پراکسی سرورهای موجود در شبکه گفته می شوند که با برقراری ارتباط با پراکسی سرور شما قادر به انجام کارهای زیر هستند:

- به اشتراک گذاری داده های ذخیره شده در مخزن کش به منظور کاهش مصرف پهنای باند و افزایش سرعت پاسخگویی به در خواستهای کاربران.
- به کارگیری پراکسی سرورهای همسایه به عنوان همکار و یا جفت جهت پاسخگویی سریع به درخواستهای کاربران.
- به کارگیری آنها جهت برقراری ارتباط از نوع همسایه و همکار .

معمولا در شبکه های متوسط و بزرگ با استفاده از چند سرور پراکسی می توان ترافیک شبکه را میان آنها تقسیم کرده و با به اشتراک گذاری منابع خود می تواند درخواستهای کاربران را در کمترین زمان ممکن پاسخ گویند.در ادامه به معرفی برخی از دستوراتی که به منظور برقراری ارتباط اسکوئید با سایر پراکسی سرورهای شبکه به کار می رود می پردازیم.

تعریف و اعلان پراکسی سرور های همسایه و همکار

دستور cache_peer برای معرفی کش سرورهایی که در همسایگی اسکوئید وجود دارند به کار می رود در واقع برای برقراری ارتباط نخست باید سرورهای مورد نظر به هم معرفی شوند، شکل کلی این دستور به صورت زیر است:

```
cache_peer HOSTNAME_OR_IP_ADDRESS TYPE PROXY_PORT  ICP_PORT  [OPTIONS]
```

در این کد در قسمت hostname_or_ip_address آدرس ip یا نام میزبان سرور مورد نظر قرار می گیرد، در قسمت TYPE نوع ارتباط تعیین می شود که میتواند از نوع sibiling،multicast یا parent باشد.در بخش icp_port هم درگاه پروتکل icp که قبلا تعیین شده است قرار می گیرد و به جای options هم تنظیماتی که می توان در نوع ارتباط تعیین کرد مشخص می شود. به مثال زیر توجه کنید:

```
cache_peer parent.example.com parent 3128 3130 default proxy-only

cache_peer 10.10.10.25 parent 3128 3130 default  proxy-only
```

در اینجا آدرس پراکسی سرور همسایه یک بار به صورت domain و یک بار بصورت ip تعریف شده است، parent نوع ارتباط را مشخص می کند و ۳۱۲۸ نام درگاه http و ۳۱۳۰ هم شماره درگاه icp است همچنین default بدین معنی است که این پراکسی سروربه صورت پیش فرض با این پراکسی سرور در ارتباط خواهد بود همچنین برچسب proxy-only هم بدین معنی است که درخواست هایی که توسط پراکسی سرور پاسخ داده می شود ذخیره (کش) نخواهد شد این گزینه زمانی کاربرد دارد که مایل به تکرار ذخیره ی داده ها در سرورها نباشیم و معمولا این دستور زمانی به کار می رود که نوع ارتباط میان سرورها از نوع پرسرعت و پهن باند باشد.نکته ی مهمی که باید به آن اشاره نمود این است که حتما قبل از ایجاد ارتباط از کافی بودن مجوزهای دسترسی در سرورها اطمینان حاصل نمائید.در فصل هشتم در مورد چگونگی ارتباط پراکسی سرورها با یکدیگر به طور مفصل بحث خواهد شد.

کنترل سریع بر آدرس های قابل دسترسی توسط سرورهای همکار

در صورتکیه تعداد سرورهای همسایه ای که پراکسی سرورشما با آن ها در ارتباط است محدود باشند در این صورت ممکن است مایل باشید بر آدرس وب سایتهایی که به سوی این سرورها ارجاع داده می شوند نظارت و کنترل بیشتری داشته باشید دستور cache_peer_domain یک راهکار سریع برای این منظور است.ساختار کلی این دستور به صورت زیر است:

```
cache_peer_domain CACHE_PEER_HOSTNAME [!]DOMAIN1 [[!]DOMAIN2 ...]
```

در این کد CACHE_PEER_HOSTNAME آدرس ip و یا نام کش سرور همسایه ای که قبلا تعریف توسط دستور cache_peer تعریف شده است را تعیین می کند با استفاده از این دستور می توان آدرس وب سایتهایی را که می خواهیم توسط این کش سرورها پاسخ داده شوند را مشخص کنیم در صورتکیه در ابتدای هر یک از نام های دامین علامت (!) قرار داده شود ،از کش سرور مورد نظر جهت پاسخگویی به آن آدرس استفاده نخواهد شد.فرض کنیم می خواهیم از کش سرور همسایه به نام videoproxy.example.com جهت بازدید از وب سایتهایی چون یوتوب ، نت فیکس و.. استفاده کنیم در

این صورت تنظیمات پیکربندی به صورت زیر خواهد بود:

```
cache_peer_domain videoproxy.example.com .youtube.com .netflix.com
cache_peer_domain videoproxy.example.com .metacafe.com
```

این تنظیمات اسکوئید را مجاب به استفاده از کش سرور همکار یا همسایه به نشانی videoproxy.example.com به منظور پاسخگویی و ارجاع درخواستهای دریافت شده از وب سایتهای youtube.com، netflix.com و metacafe.com می کند و سایر درخواستهای مربوط به سایر آدرسها به سوی این سرور همسایه ارسال نخواهند شد در واقع این سرور همسایه فقط به درخواستهایی که شامل آدرس های یاد شده باشند پاسخ می دهد.

کنترل پیشرفته دسترسی به سروهای همکار

در قسمت قبل با دستور cache_peer_domainآشنا شدیم همانطور که واضح است سطح عملکرد این دستور بسیار ابتدایی بوده و از انعطاف پذیری بالایی ندارد که در کنار دستور cache_peer_access ایجاد شده است که با بهره گیری از Acl ها از انعطاف پذیری بسیار خوبی جهت کنترل و مدیریت بر سرورهای همسایه برخوردار است.ساختار کلی آن بسیار شبیه سایر عبارات کنترلی است.ساختار کلی این دستور به صورت زیر است :

```
cache_peer_access CACHE_PEER_HOSTNAME allow|deny [!]ACL_NAME
```

برای درک بهتر این موضوع به مثال زیر توجه کنید:

```
acl my_network src 192.0.2.0/24
acl video_sites dstdomain .youtube.com .netflix.com .metacafe.com
cache_peer_access acadproxy.example.com allow my_network video_sites
cache_peer_access acadproxy.example.com deny all
```

در اینجا فقط کاربران موجود در محدوده آدرس192.0.2.0/24مجوز دسترسی به سرور همسایه acadproxy.example.com جهت دسترسی به وب سایتهای یوتیوب و متاکاف و نت فیکس را دارند.همانطور که مشاهده می کنید این روش یکی از بهترین راهکارها جهت کنترل و مدیریت دسترسی به وب سایتها و همچنین کش سرورها همسایه از سوی کاربران مختلف است . مزیت این روش آن است که شما می توانید سطح عملکرد این دستور را با ایجاد Acl ها مختلف گسترش دهید.

ذخیره سازی درخواستهای کاربران

تا اینجا بارها و بارها در مورد کش کردن محتوای وب و ذخیره سازی اطلاعات توسط اسکوئید سخن به میان آوردیم و این که این عمل موجب کاهش زمان بارگذاری صفحات وب و همچنین افزایش سرعت تبادل اطلاعات می شود اما در اینجا میخواهیم بدانیم که اسکوئید واقعا چگونه این کار را انجام میدهد. سرویس دهنده اسکوئید از حافظه رم و دیسک سخت (Hard Disk) به منظور ذخیره سازی اطلاعات استفاده می کند. ذخیره سازی اطلاعات پروسه ای پیچیده است اما اسکوئید آن را با دقت و قدرت هر چه تمام تر انجام می دهد همچنین توسط فایل پیکربندی خود یعنی squid.conf به ما این امکان را می دهد که این پروسه را به طور کامل کنترل و مدیریت کنیم به طوریکه ما می توانیم تعیین کنیم چه چیزهایی و برای چه مدت زمانی در حافظه ی دیسک سخت و یا رم ذخیره شوند و چه چیزهایی حذف شوند،همچنین میزان فضای ذخیره سازی و نوع آن نیز توسط ما تعیین می شود.حال نگاهی به مهمترین دستوراتی که فرایند ذخیره سازی داده ها توسط اسکوئید در ارتباط هستند می اندازیم.

استفاده از حافظه ی اصلی به منظور ذخیره سازی داده ها

عملیات خواندن و نوشتن داده ها در حافظه ی رم به دلیل ساختار متفاوت آن بسیار سریعتر ازدیسک سخت است به همین دلیل ذخیره سازی اطلاعات در رم و زمان دسترسی به آن در مقایسه با دیسک سخت بسیار سریع و قابل توجه است و این امر موجب تسریع در تبادل اطلاعات میان کاربران و کش سرور می شود.یکی از محدودیت های بزرگی که در مورد حافظه های رم وجود دارد ،ظرفیت کمتر آنها در مقایسه با دیسک های سخت است به طوریکه از این حافظه ها نمی توان برای ذخیره سازی گسترده داده ها استفاده کرد .برای رفع این محدودیت ها و حداکثر استفاده از این حافظ های ارزشمند،اسکوئید فقط داده هایی را که تعداد مراجعات به آنها زیاد و یا احتمال درخواست آنها بیشتر باشد و به عبارتی پرطرفدارتر باشند را در حافظه رم نگهداری می کند و بقیه داده ها را به دیسک سخت که فضای ذخیره سازی گسترده تری دارد و در مقابل عملکرد کند تری نسبت به حافظه رم دارد منتقل می کند.

درخواستهای جاری

درخواستهای جاری به داده هایی گفته می شوند که به تازگی توسط کاربران درخواست شده اند به همین دلیل مدت زمان طولانی تری در حافظه ی رم نگهداری می شوند این عمل تا زمانی که حافظه ی رم کاملا پر شود ادامه پیدا می کند پس از آن با توجه به تعداد درخواستهای ارسال شده از سوی کاربران برای یک شیء یا داده خاص محبوبترین داده ها انتخاب شده در حافظه ی رم نگهداری می شود و سایر داده ها که از محبوبیت کمتری برخوردار هستند به حافظه ی دیسک سخت منتقل می شوند.این پروسه به طور مداوم تکرار شده و مقداری از فضای رم برای نگهداری داده های جدید همواره خالی نگه داشته می

شود .

داده های پربازدید

این داده ها و یا اشیاء همان داده های پربازدید و پرطرفداری هستند که به تعداد دفعات بیشتری نسبت به سایر داده ها توسط کاربران درخواست شده اند به همین دلیل اسکوئید به منظور دسترسی سریعتر آنها را در حافظه ی رم نگهداری می کند.این داده ها در کنار سایر داده های پربازدید در حافظه رم نگهداری می شوند و زمانی که حافظه رم ظرفیت پذیرش داده های بیشتری را نداشته باشد ، داده هایی که تعداد درخواست آنها کمتر شده به دیسک سخت منتقل شده و فضای رم را برای پذیرش داده های تازه آزاد می کند البته همواره داده هایی شانس قرارگیری در حافظه ی رم را دارند که از محبوبیت بالایی نزد کاربران برخوردار باشند.

داد های ساکن

داده های ساکن همان صفحات خطایی هستند که کاربران هنگام بروز مشکل در دریافت اطلاعات و یا مشاهده صفحات وب با آن مواجه می شوند.به طور مثال در صورتیکه یک صفحه درخواستی وب با خطای ۴۰۴ مبنی بر عدم وجود داده مورد نظر مواجه شود و کاربران دیگری هم همین صفحه را درخواست کنند در این صورت اسکوئید این صفحه را جهت بررسی رفع اشکال به طور مرتب با سرور مقصد همسان سازی می کند و درصورتکیه مشکل برطرف شده باشد صفحه ی جدید به جای این صفحه خطا جایگزین می شود. به طور کلی داده های ساکن به منظور آزاد سازی فضای رم و اختصاص آن به داده های پربازدید به دیسک سخت منتقل می شوند.

تعیین میزان حافظه اصلی به منظور ذخیره سازی اطلاعات

تاکنون مطالبی را درمورد چگونگی استفاده اسکوئید از حافظه اصلی جهت نگهداری داده ها آموختیم ،در اینجا میخواهیم نحوه تعیین میزان حافظه ی رم جهت استفاده توسط اسکوئید را بیاموزیم.به طورکلی هنگام تعیین میزان حافظه رم برای استفاده توسط کش سرور نباید مقدار آن را بسیار بزرگ یا خیلی کوچک درنظر گرفت در صورتکیه میزان حافظه رم زیادی را به اسکوئید اختصاص دهیم عملکرد سیستم عامل با مشکل مواجه می شود و حافظه لازم برای اجرای سایر پروسه های خود را دریافت نمی کند که این موضوع موجب ایجاد اختلال و تاخیر در عملکرد سیستم عامل و در نهایت فعالیت اسکوئید می شود همچنین تخصیص حافظه خیلی کم نیز موجب می شود که اسکوئید نتواند از تمام توان خود در کش کردن اطلاعات استفاده کند که این خود عملکرد اسکوئید تاثیر منفی خواهد داشت. میزان حافظه پیش فرض تعیین شده در فایل پیکربندی اسکوئید ۲۵۶ مگابایت

تعیین شده است.برای اختصاص مقدار فضای بیشتر ابتدا باید از مقدار دقیق حافظه ی سیستم مطلع شد. ابزارهای گوناگونی برای تعیین دقیق این مقدار وجود دارد که در میان آنها top و free ابزارهایی هستند که به طور پیش فرض در بسیاری از توزیع های لینوکس وجود دارد.با استفاده از دستور free -m می توان از حافظه کل سیستم و همچنین مقدار مصرف شده و باقی مانده آن آگاه شد. ابزار top نیز اطلاعات جامعی در مورد منابع سخت افزاری سیستم در اختیار شما قرار می دهد.به طور مثال در صورتیکه سرور شما از ۴ گیگابایت حافظه رم برخوردار باشد از این مقدار ۱.۵ گیگابایت را برای استفاده سیستم عامل اختصاص داده و ۲.۵ گیگابایت باقی مانده را برای استفاده اسکوئید در نظر بگیرید البته نحوه تقسیم فضا میان سیستم عامل و اسکوئید بسته به عواملی چون تعداد پروسه های فعال برروی سرور و همچنین میزان حافظه مصرفی ممکن است متفاوت باشد .برای تعیین مقدار حافظه مورد استفاده توسط اسکوئید از دستور cache_mem استفاده می شود. به مثال زیر توجه کنید:

```
cache_mem 2500 MB
```

برای تعیین مقدار حافظه می توان واحد های اندازه گیری مختلفی همچون بایت (bytes)، کیلو بایت(KB) ،مگابایت(MB) و یا گیگابایت (GB)را به کار برد که در اینجا مگابایت به عنوان واحد اندازه گیری تعیین شده است.

تعیین حداکثر حجم داده های نگهداری شده در حافظه رم

محتوای موجود در وب سایت ها اندازه های مختلفی دارند ،برخی از آنها بسیار حجیم بوده و ممکن است چندین مگابایت حجم داشته باشند داده هایی که اندازه آنها بزرگ است به سرعت حافظه را پر کرده و مانع از نگهداری داده های بیشتر در حافظه رم و دیسک سخت می شوند بدین ترتیب تعداد داده های مختلفی که می توانند در حافظه سرور نگهداری شوند کاهش می یابد که این موضوع سبب پایین آمدن عملکردکشسرور می شود. برای حل این مشکل با استفاده از دستور maximum_object_size_in_memory می توان حداکثر حجم یک داده را که مجاز به ذخیره سازی در حافظه اصلی است را تعیین کرد که این مقدار باید با توجه به ظرفیت حافظه رم و نوع اتصال مورد استفاده توسط کاربران کنند تعیین شود. به مثال زیر توجه کنید:

```
maximum_object_size_in_memory 20 MB
```

در این مثال حداکثر حجم یک داده که در حافظه رم ذخیره می شود ۲۰ مگابایت تعیین شده است و این بدان معنی است که داده هایی که حجم آنها بیش از ۲۰ مگابایت باشد در حافظه رم نگهداری نخواهند شد.

تعیین روش های ذخیره سازی داده در حافظه اصلی

در نسخه های جدید اسکوئید می توان نحوه ذخیره سازی فایلهای کش شده در حافظه را کنترل کرد که این خود موجب افزایش کارایی کش سرور می شود. از دستور memory_cache_mode به منظور چگونگی مدیریت حافظه کش استفاده می شود. جدول ۲-۱ به معرفی و توضیح هرکدام از این روش ها می پردازد.

متد	توضیحات
always	این حالت که پیش گزیده اسکوئید نیز هست،تمامی اطلاعاتی که به تازگی توسط کاربران فراخوانی شده اند را در حافظه کش نگهداری می کند البته این میزان ارتباط مستقیمی با میزان ظرفیت رم و فضای اختصاص یافته به اسکوئید دارد.
disk	در این حالت داده هایی که در دیسک سخت نگهداری شده و به نسبت بیشتری مورد توجه ودرخواست کاربران واقع شده اند و دارای برچسب HTT هستند به حافظه ی کش فراخوانی می شوند.دراین روش داده ها از دیسک سخت به سوی حافظه رم فراخوانی می شوند.
network	در این حالت تنها اطلاعاتی که در جریان ارتباط میان کش سرورهای همسایه مبادله شده در حافظه کش نگهداری می شوند به کارگیری این حالت تنها زمانی مفید است که اسکوئید با همسایه های خود در شبکه در ارتباط بوده و با آنها تبادل اطلاعات داشته باشد.

برای به کارگیری این دستور کافی است هرکدام از روش های بالا را همراه با دستور memory_cache_mode به صورت زیر به کار ببرید:

```
memory_cache_mode always
```

در این مثال روش always انتخاب شده است یعنی داده هایی که به تازگی توسط کاربران فراخوانی شده اند در حافظه کش نگهداری می شود.

استفاده از دیسک سخت برای ذخیره سازی داده ها

تاکنون نحوه به کارگیری حافظه رم جهت نگهداری و کش کردن درخواستهای کاربران را بررسی کردیم امّا همانطور که پیش تر هم گفته شد حافظه های رم به دلیل حجم محدود آنها در ذخیره سازی داده ها و همچنین موقتی بودن نگهداری داده ها در آنها استفاده از یک فضای ذخیره سازی دائمی و با ظرفیت بالا در کنار آنها را به امری اجتناب ناپذیر مبدل کرده است .امروزه فضای ذخیره سازی دیسک های سخت پیشرفت قابل توجهی داشته اند .به طوریکه به کارگیری دیسک های سخت با برخورداری از چند صد گیگابایت ظرفیت امری عادی محسوب می شود.استفاده از دیسک های سخت با سرعت خواندن و

نوشتن بالا نظیر دیسک های جامد یا SSD و همچنین دیسک های مکانیکی با سرعت بالا بر کیفیت کش سرور تاثیر مستقیمی دارند به طوریکه هرچه زمان تاخیر انتقال داده ها کاهش یابد پاسخ به درخواستها سریعتر شده و احتمال به وجود آمدن وقفه ها در شبکه به طور چشمگیری کاهش می یابد.در ادامه به بررسی چگونگی به کارگیری دیسک سخت به منظور کش کردن داده ها می پردازیم.

تعیین میزان فضای ذخیره سازی

دستور cache_dir برای تعریف فضای دیسک سخت جهت به کارگیری توسط اسکوئید به کار می رود و اسکوئید از آن برای ذخیره سازی اطلاعات استفاده می کند. ساختار کلی این دستور به صورت زیر است:

cache_dir STORAGE_TYPE DIRECTORY SIZE_IN_Mbytes L1 L2 [OPTIONS]

آشنایی با انواع فضاهای ذخیره سازی در لینوکس

تمامی سیستم عاملهای موجود از سیستم فایل ها (File systems) به منظور ایجاد بستری مناسب جهت ذخیره سازی داده ها در فضای دیسک سخت بهره می برند.درواقع بدون وجود یک سیستم فایل ،امکان ذخیره سازی ،تغییر و یا حذف اطلاعات برروی دیسک سخت غیر ممکن است . سیستم فایل ها انواع مختلفی دارند که بسته به نوع سیستم عامل متفاوت اند همچنین ساختار آنها نیز بعضا با هم متفاوتند.در دنیای لینوکس و یونیکس سیستم فایل های مختلفی وجود دارد که مهمترین آنها عبارتند از: ext2, ext3, ext4, reiserfs, xfsوUFS که معمولا در تمامی توزیعهای مشهور پشتیبانی می شوند. سیستم فایل ها برخی از توابع سیستمی مانند open(), close()وread () را فراخوانی می کنند همچنین با فراهم ساختن بستری مناسب امکان مدیریت داده ها توسط سایر نرم افزارها را نیز فراهم می کنند .اسکوئید هم مانند سایر نرم افزارها از این توابع جهت ارتباط با سیستم فایلها و مدیریت فایلهای کش شده بهره می گیرد.به منظور تسریع در فرایند مدیریت اطلاعات و همچنین مدیریت بهتر فایلها اسکوئید از یک ساختار ذخیره سازی اختصاصی که برروی سیستم فایل های اصلی ساخته می شوند استفاده می کند درواقع این ساختار مشابه یک سیستم فایل که منحصرا جهت به کارگیری توسط اسکوئید طراحی شده عمل می کند. ufs,aufsو diskd از جمله انواع مختلف این سیستم فایلها به شمار می روند .دراین میان ufs ساده ترین نوع آنها به شمار می رود و تمامی ورودی و خروجی های داده (I/O) را به وسیله اسکوئید پردازش ومدیریت می کند.از ufs بیشتر در سرورهایی که بارترافیکی کمتری دارند استفاده می شود و به کارگیری آن در محیط هایی که نیاز به پردازش و تبادل داده بالایی دارند مناسب نیست و موجب ایجاد وقفه های بعضا طولانی در فرایند انتقال اطلاعات می شود که این امر موجب کاهش شدید کارایی می شود. aufs در واقع نسخه اصلاح شده ufs با قابلیت پشتیبانی از پردازش نا متقارن (asynchronous) است . به عبارت

دیگر aufs همان ufs است که قابلیت پردازش متقارن اطلاعات به آن افزوده شده است که در واقع تنها مشکل عمده ufs نیز به شمار می رود . پردازش نا متقارن اطلاعات باعث جلوگیری از ایجاد وقفه های طولانی در پردازش اطلاعات توسط اسکوئید می شود. در واقع این ساختار باعث ایجاد نوعی صف بندی در دریافت درخواستها می شود و بدین ترتیب اسکوئید می تواند تعداد درخواستهایی را که نمی تواند به طور همزمان پاسخ دهد را در صف نگهداری کند .در محیط هایی که میزان ترافیک شبکه بالاست استفاده از aufs به جای ufs کاملا منطقی به نظر می رسد.

نوع دیگری از ساختار ذخیره سازی در اسکوئید Diskd نام دارد که ساختاری مشابه aufs دارد با این تفاوت که به جای رشته ها از پروسه ای خارجی برای انجام عملیات ورودی و خروجی داده (I/O) استفاده می کند.برخلاف موارد قبلی در این نوع برای هر شاخه ی cache_dir یک صف بندی کاملا جدا به کار می رود و در آن پیام هایی جهت برقراری ارتباط با اسکوئید ایجاد می شود .این نوع ساختاردر برخی موارد که میزان بار پردازشی بر پراکسی سرور زیاد است موجب سرریز شدن پراکسی سرور می شود.به هر حال برای ایجاد یک شاخه دایرکتوری از نوع diskd به صورت زیر عمل می کنیم:

cache_dir diskd DIRECTORY SIZE_Mbytes L1 L2 [OPTIONS] [Q1=n] [Q2=n]

یکی از گرینه های اضافی ای که در این نوع به چشم می خورد ،Q1 و Q2 هستند و مقادیر آنها تعیین کننده درخواستهایی است که اسکوئید نتوانسته است به طور مستقیم به آنها پاسخ دهدکه این درواقع همان پردازش خارجی ای است که به آن اشاره شد.مقدار پیش فرض برای Q1،64 و برای Q2،72 در نظر گرفته شده است و هنگامی که این فضا ها پر شوند اسکوئید به نوعی به حالت تعلیق درآمده و هیچگونه درخواستی را پاسخ نمی دهد و و زمانی یکی از فضاهای مورد نظر خالی شد پیامی مبنی بر وجود فضای آزاد به سوی اسکوئید فرستاده می شود و بدین ترتیب امکان پردازش درخواستهای جدید و پاسخگویی به آنها فراهم می شود .با توجه به توضیحات بالا در صورتیکه مقدار Q1 بیش تر از Q2 باشد اسکوئید به حالت block mode وارد می شود به این معنی که ابتدا به درخواستهایی که در صف وجود دارند پاسخ می دهد در این حالت در صورتیکه تمام فضای موجود در صف بندی پر شود باعث ایجاد تاخیر در پاسخگویی اما (HIT Rate) بالاتر می شود که موجب صرفه جویی بیشتر در مصرف پهنای باند و منابع می شود در مقابل زمانی که مقدار Q2 بیشتر از Q1 در نظر گرفته شود،اسکوئید ابتدا به جمع آوری درخواستهای شبکه و قرار دادن آنها در صف اقدام می کند ، در این حالت پردازش داده ها سریعتر اما میزان HIT rate به طور قابل توجهی کاهش می یابد. با مقایسه این ساختار ها می توان گفت که ufs ساختاری ابتدایی و قدیمی ، aufs ساختاری جدید و هوشمند و diskd قابلیت سفارشی سازی بیشتری دارد تنها تفاوتی که می توان میان aufs و diskd قائل شد نوع مدیریت منابع پردازشی است که در aufs کاملا پویا و در diskd قابل کنترل و مدیریت است.

نام گذاری و ساخت دایرکتوری های کش سرور

برای ساخت دایرکتوری های کش سرور می توان هر قسمتی از سیستم فایل را در نظر گرفت .اسکوئید به طور پیش فرض مسیر های /var/spool/squid و /usr/local/squid/var/cache را برای ساخت دایرکتوری های خود برگزیده است که این مسیرها را می توان با مکانهای دلخواه خود جایگزین کرد و پس از آن اسکوئید کلیه مراحل ساخت دایرکتوری را انجام می دهد توجه به این نکته ضروری است برای اینکه اسکوئید بتواند عملیات خواندن و نوشتن را در مخزن کش انجام دهد باید مجوزهای لازم جهت خواندن و نوشتن داده ها به آن اعطا شود که این امر با دستور chown امکان پذیر است.همچنین اسکوئید دایرکتوریهای را که قبلا ساخته شده اند را دوباره سازی نخواهد کرد.در صورتیکه بیش از یک دیسک سخت در اختیار دارید و می خواهید تمام فضای یکی از آن ها را به مخزن کش اختصاص دهید مانند مثال زیر عمل کنید:

```
$ mkdir /drive/
$ mount /dev/sdb1 /drive/squid_cache/
$ mkdir /drive/squid_cache
$ chown squid:squid /drive/squid_cache/
```

دراین مثال نام دیسک درایو دیگری که برای این کار اختصاص داده ایم /dev/sdb است و یکی از پارتیشن های آن را که می خواهیم دایرکتوری های کش روی آن ساخته شود /dev/sdb است.با استفاده از دستور mount آن را فراخوانی می کنیم .در اینجا نام دایرکتوری مورد نظر ما squid_cache است در مرحله ی بعد با اجرای دستور chown squid:squid //drive/squid_cache مجوزهای های لازم برای خواندن و نوشتن به آن اعطا می شود و در پایان مسیر مورد نظر را در مقابل برچسب cache_dir مشخص می کنیم .بدین تریب می توان از یک دیسک سخت کاملا مجزا برای ساخت دایرکتوریهای مخزن کش استفاده کرد مزیت این روش این است که فضای ذخیره سازی کاملا جدا از سیستم عامل بوده و هرگونه اشکال در دیسک مانع از کارافتادن کل سرور می شود دیگر اینکه همواره می توان اطمینان داشت در صورتیکه فضای کل دیسک سخت هم تمام شود سرور و سیستم عامل به فعالیت خود ادامه خواهند داد به علاوه طبیعتا فضای بیشتری هم به کش سرور جهت ذخیره سازی فایلهای کش شده اختصاص داده می شود.

تعیین حجم مخزن کش سرور

یکی از نکات حائز اهمیت در این بخش آن است که همواره جهت تعیین مقدار حافظه ی دیسک سخت به منظور ساخت مخزن کش سرور باید از واحد اندازه گیری مگابایت (MB)استفاده کرد.به طور مثال اگر میزان فضای درنظر گرفته 100 گیگابایت باشد برای تبدیل آن به مگابایت باید مقدار عددی به گیگابایت را در 1024 ضرب کنید مثلا در اینجا با یک عمل ضرب ساده عدد 102400 مگابایت به دست می آید که معادل 100 گیگابایت است. برای تعیین سایر مقادیر هم مانند روش بالا عمل کنید. در صورتیکه کل فضای یک دیسک سخت را برای ساخت مخزن کش سرور در نظر گرفته اید حدود 5 تا 15 درصد از کل فضای آن را به فایلهای موقت اسکوئید اختصاص دهید.

تعیین تعداد دایرکتوری های سطح اول و دوم

اسکوئید برای ساخت دایرکتوری های کش سرور از یک ساختار سلسله مراتبی بهره می گیرد در این روش از دو سطح دایرکتوری به نام های L1 و L2 استفاده می شود دلیل این کار جستجو و دسترسی سریعتر به داده ها عنوان شده است.در این روش در داخل هریک از دایرکتوری های سطح اول تعدادی دایرکتوری سطح دوم ساخته می شود که تعداد آن ها بسته به مقادیر تعیین شده برای L1 و L2 متغیر است. در ادامه با نحوه ساخت و مقدار دهی آن ها بیشتر آشنا خواهیم شد.

کش سرور در حالت فقط خواندنی

یکی ازدستوراتی که از آن برای جلوگیری از حذف و یا نوشتن داده های جدید به کار می رود no-store است.در صورت بکارگیری این دستور همراه با cache_dir اطلاعات موجود در مخزن کش به صورت فقط خواندنی در می آید و امکان هرگونه بروزرسانی ، حذف و یا اضافه ی اطلاعات از اسکوئید سلب می شود اما امکان دسترسی به داده های قدیمی که پیش از اجرای این دستور در مخزن ذخیره شده است وجود دارد. استفاده از این دستور جز درموارد ضروری اصلا توصیه نمی شود زیرا اطلاعات کش شده را در یک حالت کاملا ساکن قرار می دهد و امکان بروزرسانی در آنها وجود ندارد.

اضافه کردن دایرکتوریهای مخزن کش

برای اضافه کردن یک دایرکتوری جدید و معرفی آن به اسکوئید از دستور زیر استفاده می شود:

```
cache_dir aufs /squid_cache/ 51200 32 512
```

در این مثال مسیر دایرکتوری ها در /squid_cache بوده و میزان فضای در نظر گرفته شده برای آن ۵۰ گیگابایت است همچنین تعداد دایرکتوری های سطح اول ۳۲ و تعداد دایرکتورهای سطح دوم ۵۱۲ عدد تعیین شده است.بنابراین اگر فرض کنیم حجم میانگین هر یک از اشیاء ذخیره شده درفضای مخزن کش سرور ۱۶ کیلوبایت باشد در این صورت با انجام یک محاسبه ساده ریاضی به صورت 51200*1024÷(32*512*16) = 200 به این نتیجه می رسیم که در هر دایرکتوری سطح دوم تعداد ۲۰۰ فایل مختلف ذخیره می شوند که مقدار مناسبی به نظر می رسد البته در اینجا ۱۶ کیلوبایت یک مقدار فرضی است. بدین ترتیب با توجه به میزان فضای ذخیره سازی موجود در دیسک سخت این مقدار می تواند افزایش و یا کاهش یابد.

بهینه سازی دایرکتوری های کش

در صورتیکه چند دایرکتوری جداگانه در قسمت دایرکتوری های کش سرور تعریف شده باشد استفاده از یک الگوریتم تقسیم بار به منظور تقسیم بار شبکه در شرایطی که ترافیک برروی پراکسی سرور زیاد است موجب افزایش کارایی و کاهش تاخیر در پاسخ به درخواستها می گردد.در واقع هدف از این کار به کارگیری تمامی دایرکتوری ها و جلوگیری از اعمال بار اضافی به قسمت های مشخصی از دایرکتوری ها است.از دستور store_dir_select_algorithm به منظور انتخاب بهترین الگوریتم تقسیم بار استفاده می شود که در واقع به عنوان راهی برای استفاده حداکثری از تمامی دایرکتوری ها به شمار می رود.گزینه های مورد استفاده همراه این دستور least-load و round-robin هستندو ساختار کلی دستور به صورت زیر است:

```
store_dir_select_algorithm least-load|round-robin
```

در صورتیکه بخواهیم تمامی داده های کش شده به طور مساوی در تمام دایرکتوریهای موجود تقسیم شود گزینه مورد نظر round-robin خواهد بود اما در صورتیکه بازدهی بالا و زمان تاخیر کم مد نظر است least-load گزینه مناسب است که با انتخاب این دستور اسکوئید دایرکتوریهایی راکه کمترین بار پردازشی بر آنها اعمال شده باشد را جهت پذیرش داده های بیشتر انتخاب می کند.

تعیین حجم فایلهای ذخیره شده

تعیین حجم برای فایلهای کش شده یکی از مهمترین نکاتی است که در کارایی و عملکرد کش سرور تاثیر بسیار زیادی دارد.به طور کلی در صورتکیه حجم داده های ذخیره شده در دیسک بسیار بزرگ باشد، باعث پر شدن سریع فضای دیسک سخت و در نتیجه کاهش تعداد داده های کش شده توسط اسکوئید می شود که خود موجب کاهش کارایی و سرعت بازخوانی داده ها می شود ، به طور مثال در صورتکیه کل فضایی که به کش سرور اختصاص داده شده ۱۰ گیگابایت باشد و حداکثر حجم داده های ذخیره شده نیز ۵۰۰ مگابایت تعیین شده باشد با یک محاسبه ساده می توان دریافت تعداد فایلهایی که می توانند در این فضا ذخیره شوند بسیار محدود هستند که این عامل باعث کاهش تعداد داده های ذخیره شده در مخزن کش سرور می شود. برای جلوگیری از بروز این گونه مشکلات دستورات minimum_object_size و maximum_object_size پیش بینی شده اند که به ترتیب برای تعیین حداکثر و حداقل حجم داده های کش شده به کار می روند.کمترین مقدار به طور پیش فرض صفر درنظر گرفته شده است که به معنی عدم وجود محدودیت برای فایلهای با حجم پایین است. تعیین حداکثر حجم هم بسته به میزان فضای ذخیره سازی و همچنین میزان پهنای باند موجود متغیر است به طوریکه اگر سرور و کاربران از اتصالات پرسرعت استفاده می کنند مقدار آن را می توان افزایش داد و در غیر اینصورت افزایش آن معقول به نظر نمی رسد. شکل کلی به کارگیری این دستورات به صورت زیر است:

```
minimum_object_size 0 KB
```

```
maximum_object_size 96 MB
```

در این مثال کمترین حجم برابر صفر است و به معنی عدم وجود محدودیت برای فایلهای کوچک است همچنین حداکثر حجم داده های قابل ذخیره سازی ۹۶ مگابایت است یعنی فایلهای بزرگتر از ۹۶ مگابایت در مخزن کش سرور ذخیره نخواهند شد.

تعیین سقف مجاز برای حذف جایگزینی داده های قدیمی

طی یک دوره زمانی فضای اختصاص داده شده به مخزن کش سرور به تدریج تمام می شود از طرفی میزان قابل توجهی از اطلاعات موجود در مخزن کش سرور قدیمی بوده و کاربران معمولا دیگر به آنها نیازی ندارند و فراخوانی نمی شوند.هنگامی که فضای درنظر گرفته شده برای مخزن کش از یک مقدار مشخص که توسط ما تعیین می شود فراتر رود اسکوئید شروع به حذف فایلهای قدیمی می کند که این مقدار مشخص توسط دستورات cache_swap_low و cache_swap_high تعیین و مقادیر ۰ تا ۱۰۰ را شامل می شوند.شکل کلی دستورات مانند مثال زیر است:

```
cache_swap_low 96
cache_swap_high 97
```

در اینجا در صورتیکه بیش از ۹۶ از کل فضای اختصاص یافته به مخزن کش سرور مصرف شده باشد، اسکوئید شروع به حذف فایلهای قدیمی می کند تا فضای مصرف شده را به حدود ۹۶ درصد برگرداند . گاهی اوقات به دلیل فشار کاری شدید و مصرف سریع فضای آزاد مقدار فضای مصرفی بازهم بالاتر می رود در اینجا دستور دوم وارد عمل شده و با حذف مکرر داده ها میزان فضای آزاد را به حدود تعیین شده در دستور دوم که در اینجا ۹۷ درصد تعیین شده بر می گرداندکه این عمل از طریق حذف مکرر داده ها تا رسیدن به سقف مجاز ذخیره سازی ادامه می یابد.مقدار پیش گزیده ی این دستورات به ترتیب ۹۰ و ۹۵ درصد است که بیشتر برای فضاهای ذخیره سازی محدود مانند ۱۰ گیگابایت مناسب است اما در صوریتکه میزان فضای ذخیره سازی اختصاص یافته به کش سرور زیاد است (مثلا ۵۰ گیگابایت و یا بیشتر) می توان این مقادیر را بیشتر کرده تا فضای بیشتری برای ذخیره سازی اطلاعات مورد استفاده قرار گیرد.

سیاست های حذف داده ها

همانطوری که در بالا اشاره شد به دلیل ذخیره سازی مداوم اطلاعات در دیسک سخت این فضا به سرعت مصرف شده در حالیکه نیاز به فضای ذخیره سازی همچنان وجود دارد.اسکوئید به منظور حذف برخی از اطلاعات قدیمی و جایگزینی آنها با داده های جدید از سیاست های کلی ای بهره می گیرد که در ادامه به معرفی آنها می پردازیم.

راهکار Least recently used (LRU)

lru ، ساده ترین روش حذف و جایگزینی داده ها به شمار می رود.در این روش داده ها و اشیاء قدیمی ای که زمان بیشتری از آخرین فراخوانی آنها توسط کاربران سپری شده باشد را حذف و با داده های جدید جایگزین می کند همچنین روش جایگزینی داده ها با استفاده از لیست ساختاری داده ها صورت می گیرد. این روش به طور پیش فرض توسط اسکوئید به کار می رود.

راهکار Greedy dual size frequency (GDSF)

در این روش اسکوئید تلاش می کند داده هایی که پرطرفدار بوده و حجم کمتری دارند را نگهداری کرده و داده های با حجم بالا را علی رغم اینکه پربازدید هستند حذف می کند تا فضای بیشتری برای ذخیره سازی داده های جدید آزاد شود .درواقع هدف از انجام این کارها نگهداری داده های کم حجم و پربازدید و حذف داده های حجیم است به طوریکه در صورتکیه دو شیء با میزان محبوبیت برابر وجود داشته باشند اسکوئید ، شیء ای را که حجم کمتری دارد را نگهداری کرده و داده حجیم را حذف می کند. در این روش میزان HIT rate بیشتر شده اما در مصرف پهنای باند صرفه جویی چندانی نمی شود.

راهکار Least frequently used with dynamic aging (LFUDA)

برخلاف مورد قبل تاکید این روش بر صرفه جویی بیشتر در مصرف پهنای باند در مقایسه با HIT Rate است در این روش تمامی داده های پرطرفدار اعم از داده های کوچک و بزرگ نگهداری شده و داده هایی که از بازدید کمتری برخوردار بوده اند حذف می شود. با توجه به نوع ساختار به کاررفته در LFUDA میزان صرفه جویی در مصرف حافظه نبست به موارد قبلی کمتر است. در صورت به کارگیری این روش، حداکثر حجم داده های قابل ذخیره سازی که در دستور maximum_object_size تعیین می شود را باید بسته به میزان فضای حافظه تعیین کرد که طبیعتا با توجه به مکانیسم مورد استفاده در این روش که تاکید آن بر صرفه جویی در مصرف پهنای باند است باید حداکثر حجم داده های ذخیره شده در کش بزرگتر از سایر روش ها انتخاب گردد. پس از انتخاب گزینه مناسب میان مواردی که ذکر شد با استفاده از دستور ات memory_replacement_policy برای حافظه رم و cache_replacement_policy برای حافظه دیسک سخت می توان این سیاست ها را پیاده سازی نمود همچنین موارد انتخاب شده برای حافظه رم و دیسک سخت می توانند متفاوت باشند. به مثال زیر توجه کنید:

```
memory_replacement_policy  lru
cache_replacement_policy  LFUDA
```

دراین مثال مکانیسم انتخابی حذف داده ها برای حافظه رم lru و برای دیسک سختLFUDA تعیین شده است.

بهینه سازی اسکوئید

گرچه با به کارگیری تنظیمات پیش فرض اسکوئید نیز میزان بازدهی بسیار مناسب است اما با انجام برخی تنظیمات پیشرفته تر می توان این بازده باز هم افزایش داد. این تنظیمات پیشرفته شامل جلوگیری از کش شدن درخواستهای کاربران خاص،برخی از محتوای موجود در وب سایتها و نظارت بیشتر بر محتوای کش شده در سرور می باشد که نتیجه آن صرفه جویی بیشتر در مصرف پهنای باند و کاهش میزان تاخیر در پاسخگویی به درخواستهای کاربران است.

سفارشی کردن ذخیره سازی داده ها

اسکوئید به طور پیش فرض درخواستهای تمامی کاربرانی که به شبکه دسترسی دارند را ذخیره می کند. در صورتیکه بخواهیم داده های برخی از این کاربران ذخیره نشود می توان با مشخص کردن آدرسهای IP آنها از کش شدن محتوای جستجوی آنها در سرور جلوگیری کنیم.یکی از مواردی که استفاده از این روش را ضروری می نماید ، جلوگیری از کش شدن محتوای موجود برروی سرورهای شبکه داخلی سازمان است که با توجه به کیفیت و سرعت بسیار مطلوب شبکه های محلی ،ذخیره سازی داده های موجود برروی این سرورها حاصلی جز هدر رفتن فضای دیسک سخت کش سرور نخواهد داشت.با استفاده از دستور cache می توان بر ذخیره سازی داده ها نظارت بیشتری داشت همچنین نحوه به کارگیری آن بسیار آسان است.به مثال زیر توجه کنید:

```
acl local_machines dst 192.0.2.0/24 198.51.100.0/24
cache deny local_machines
```

در این کد، درخواستهای کاربرانی که در محدوده آدرس 192.0.2.0/24 و 198.51.100.0/24 قرار دارند کش نخواهد شد و تمامی درخواستهای آنان به طور مستقیم به سرورهای مقصد که در واقع همان سرورهای محلی و داخل سازمانی است ارجاع داده خواهند شد.

آشنایی با Refresh Pattern ها

refresh_pattern ها به مظنور نظارت مستقیم بر وضعیت داده های ذخیره شده ایجاد شده اند .با ساتفاده از این دستورات می توان به طور مستقیم بر نوع داده هایی که میخواهیم کش شوند و همچنین زمان ماندگاری و بروزآوری هرکدام نظارت داشته باشیم. به کارگیری refresh_pattern ها موجب بروزرسانی هوشمند داده ها و جایگزینی خودکار آنها با داده های قدیمی می شود در واقع استفاده مناسب و به جا از این دستورات باعث می شوند اطلاعاتی که کاربران دریافت می کنند همواره بروز باشد در حالیکه Acl هایی مانند cache فقط می توانند تعیین کنند که چه داده هایی ذخیره و چه داده هایی ذخیره نشوند. اهمیت این موضوع در مورد وب سایتهایی که محتوای موجود در آنها مدام درحال تغییر است مانند پایگاههای خبری

بسیار زیاد است .نحوه عملکرد refresh pattern ها شبیه ACL هاست به این تفاوت که نحوه عملکرد refresh pattern ها هوشمندانه تر و دامنه فعالیت گسترده تری نسبت به ACL ها دارند و کارهایی را که با استفاده از ACL ها سخت و بعضا ناممکن هستند به وسیله refresh pattern ها به خوبی قابل انجام است. ساختار کلی این دستورات به صورت refresh_pattern [-i] regex min percent max [OPTIONS] است . به جای پارامتر regex یک عبارت منطقی و منظم مانند URLقرار می گیرد هر خط از یک refresh_pattern می تواند طیف وسیعی از آدرس وب سایتهای مختلف را شامل شود.درحالت پیش فرض refresh pattern ها به کوچک و بزرگ بودن حروف حساس هستند برای غیر فعال کردن این حالت می توان از پسوند i- همراه با دستورات استفاده کرد. برخی از داده های موجود برروی وب سرورها زمان انقضا و بروزآوری آنها مشخص نیست ودر نتیجه قابلیت بروزرسانی هوشمند را ندارند به همین دلیل در صورت بروزرسانی ، کش سرور از این موضوع مطلع نمی شود و معمولا موجب بروز مشکلات متعددی می شود.برای حل این مشکل با استفاده از برچسب min می توان مدت زمانی را که پس از آن داده ها باید بروز شوند و اطلاعات موجود در کش با داده های اصلی موجود برروی وب سرور های مقصد همسان شود تعیین نمود .مقدار پیش فرض این گزینه صفر است دلیل آن هم جلوگیری از ایجاد مشکلات در نمایش وب سایتهای پویاست.در صورتکیه از وضعیت نمایش وب سایت مورد نظر اطمینان کافی دارید می توانید این مقدار را افزایش دهید.پارامتر percent ، مدت زمانی را که کش سرور پس از آن یک داده را بروز می کند را مشخص می کند، این برچسب در غیاب هدر Expires به کار گرفته می شود.مدت زمان بروزآوری یک داده توسط هدر هایی نظیر Date و Last-Modified تعیین می شود بنابراین اگر مقدار تعیین شده برای برچسب ۵۰percent و تفاوت زمانی میان Last-Modified و Date یک ساعت باشد باشد داده مورد نظر هر ۳۰ دقیقه یک بار به روزرسانی می شود.به طور مشابه ، برچسب های min و max نیز بیشترین و کمترین مدت زمان بروزرسانی یک داده به دقیقه را تعیین می کنند و در صورتکیه یک داده بیش از مدت زمان تعیین شده در برچسب max در مخزن کش باقی بماند آن داده در بروزرسانی های بعدی بروزآوری نخواهد شد.برای اینکه تمامی داده های موجود همواره پس از متقضی شدن اعتبار آنها دوباره اعتبار سنجی شوند وب سرور همواره از طریق هدر های HTTP مدت عمر باقی مانده یک داده را به اطلاع پراکسی سرور می رساند،که به میزان طول عمر باقی مانده یک داده اصطلاحا lm-factor گفته می شود.

توجه: مقادیر زمانی موجود در هدر **Expires HTTP** *،مقادیر تعیین شده در برچسب های* **min** *و* **max** *را غیرفعال می کند.*

به طور کلی refresh pattern ها داده ها را به یکی از صورت های زیر طبقه بندی می کنند:

- منقضی: در صورتیکه از مدت زمانی که در هدر HTTP تعیین شده ،زمان زیادی گذشته باشد.
- بروز: در صورتیکه هنوز مدت زمانی که در هدر HTTP به عنوان زمان انقضای داده تعیین شده فرانرسیده است.

- منقضی: در صورتکیه مدت زمان موجود در response age بیش از مقدار تعیین شده در برچسب Max باشد.
- بروز: اگر مقدار تعیین شده درlm-factor کمتر از مقدار percent باشد.
- بروز: درصورتیکه زمان موجود در گزینه response ageکمتر از مقدار تعیین شده در Min باشد.
- در سایر موارد یک داده به عنوان منقضی در نظر گرفته می شود.

مثال زیر نحوه عملکرد یک refresh pattern را نشان می دهد:

```
refresh_patten -i ^http://example.com/test.jpg$ 0 60% 1440
```

در اینجا فرض می کنیم در طی یک ساعت گذشته کاربری یک تصویر با فرمت jpg را از آدرس example.com درخواست کرده است و این تصویر شش ساعت پیش برروی این وب سرور قرارگرفته است همچنین فرض کنیم این وب سرور زمان انقضای آن را تعیین نکرده است با توجه به مفروضات بالا و مقادیر تعیین شده می توان گفت:

- در زمانی که کاربر این تصویر را درخواست کرده عمر این داده پنج ساعت بوده است.
- در حال حاضر زمان response age یک ساعت است.
- مقدار تعیین شده برای پارامتر lm-factor ، بیست درصد است.

همچنین برای مشخص کردن اینکه داده هنوز بروز است یا خیر با توجه به مقادیر داده شده پاسخ مثبت است زیرا زمان پاسخگویی ۶۰ دقیقه است و از زمان حداکثری آن یعنی ۱۴۴۰ کمتر است همچنین مقدار lm-factor با توجه به محاسبات ۲۰ درصد است وکمتر از مقدار حداکثری آن یعنی ۶۰ درصد است.زمان انقضای این تصویر ۳ ساعت پس از آخرین درخواست است یعنی ۲ ساعت از حالا به زمان انقضا و عمر این داده باقی مانده است.

تنظیمات مربوط به Refreh Pattern ها

در بیشتر موارد زمان انقضای تمامی داده های موجود برروی وب سرورها تعیین می شوند .برخی از داده هایی که برروی این وب سرور ها قرار دارند، مانند کدهای جاوااسکریپت،کدهای CSS همچنین سایر کدهایی که در طراحی قالب وب سایتها به کار می روند به ندرت دچار تغییر می شوند. به منظور بهره وری بیشتر از فرایند کش کردن اطلاعات با استفاده از دستوراتی که برای این کار پیش بینی شده اند می توان از منقضی شدن سریع این کدها و بروزرسانی پی درپی آنها جلوگیری کرد برای این کار می بایست هدر هایی که پیامهای انقضای داده ها را حمل می کنند نادیده گرفته شوند.در ادامه با این دستورات و نحوه به

کارگیری آنها آشنا خواهیم شد.

override-expire

این دستور ، هدر های حاوی مدت زمان داد ه ها (Expire Headers) را نادیده گرفته و از بروزشدن خودکار داده هایی که این برچسب را به همراه دارند جلوگیری می کند.هنگام استفاده از این دستور سایر پارامترهای موجود مانند Min، Max و Percent در تعیین زمان بروزرسانی داده ها توسط پراکسی سرور نقش مهمی را ایفا می کنند.به عبارت دیگر در صورت به کارگیری این برچسب ،مقادیر موجود در عبارات min و max و همچنین percent باید به دقت تعیین شوند.

override-lastmod

این دستور اسکوئید را مجاب می کند تا مقادیر موجود در هدر Last-Modified را نادیده می گیرد و اسکوئید را مجبور به استفاده از مقادیر تعیین شده در برچسب min جهت تعیین زمان بروزآوری یک داده می کند.این دستور فقط زمانی کاربرد دارد که مقدار موجود در min مقداری غیر از صفر تعیین شده باشد.

reload-into-ims

این دستور باعث می شود تمامی برچسب های no-cache در هدر های HTTP به If-Modified-Since تبدیل شوند. به کارگیری این دستور تنها زمانی مفید است که هدر Last-Modified موجود باشد.

ignore-reload

استفاده از این دستور باعث می شود تا اسکوئید عبارات no-cache و reload موجود در هدر درخواست ها را نادیده بگیرد.

ignore-no-cache

این دستور باعث می شود تا اسکوئید برچسب no-cache موجود در هدر های HTTP را نادیده بگیرد .

Ignore-no-store

هدر cache-control: no-store به کاربران اعلام می کند که داده های مورد نظرشان قابلیت ذخیره سازی را ندارند.دستور ignore-no-store موجب نادیده گرفته شدن این هدر ها و ذخیره سازی اطلاعات توسط اسکوئید می شود .

Ignore-must-revalidate

سر فایل Cache-Control: must-revalidate موجب می شود درخواستهایی که این برچسب را به همراه دارند قبل از ارجاع به کاربر توسط سرورهای مقصد دوباره اعتبار سنجی شوند که این موضوع سبب می شود هرگاه کاربری داده ای را از پراکسی سرور درخواست کند در صورتکیه پاسخ درخواست مورد نظر در مخزن کش سرور وجود داشته باشد و از نظر مدت زمان اعتبار هم مشکلی نداشته باشد اما باز هم باید مشخصات داده مذکور با سرور مقصد همسان سازی شود که این موضوع سبب افزایش ترافیک و مصرف پهنای باند خواهد شد.عبارت ignore-must-revalidate اسکوئید را مجاب به نادیده گرفتن این هدر می کند.

ignore-private

برخی از اطلاعات موجود با برچسب Cache-Control: private در گروه اطلاعات شخصی یا حساس و غیرقابل کش توسط کش سرور قرار می گیرند و کش سرور هیچ یک از اطلاعاتی را که حاوی این هدر باشند را در مخزن خود نگهداری نمی کند.استفاده از دستور ignore-private موجب نادیده گرفتن شدن این هدر توسط اسکوئید می شود.

ignore-auth

در صوررت استفاده از دستور ignore-auth تمامی درخواستها از جمله درخواستهایی را که شامل کلمات عبور کاربران هستند را ذخیره می کند استفاده از این دستور سطح امنیت اطلاعات شخصی کاربران را کاهش می دهد.

refresh-ims

این دستور یکی از مفید ترین دستورات موجود است که موجب می شود اسکوئید داده کش شده با سرور اصلی را از نظر سطح اعتبار همسان سازی کند هرگاه درخواستی با هدر If-Modified-Since از یک کاربر دریافت کند.به کارگیری این دستور موجب ایجاد تاخیر جزئی در پاسخگویی به درخواستها می شود اما در مقابل کاربران همواره داده های معتبر و بروز رادریافت می کنند.مثال زیر چگونگی به کارگیری این دستور را نشان می دهد:

```
refresh_pattern -i .jpg$ 0 60% 1440 ignore-no-cache ignore-no-store reload-into-ims
```

این کد موجب می شود تمامی تصاویر با فرمت jpg کش شوند خواه سرورهای مقصد اجازه ی این کار را داده باشند و یا نه این کد تنها زمانی اطلاعات را بروز می کند که :

- اگر هدر Expires موجود باشد و زمان انقضای داده فرارسیده باشد..
- اگر هدر Expires موجود نباشد و عمر داده مورد نظر در مخزن کش بیشتر از مقدار تعیین شده در عبارت Max شده باشد.

مثال زیر صفحه خانگی گوگل را به مدت 6 ساعت در حافظه ی کش نگهداری می کند و پس از آن اقدام به بروزرسانی داده های کش شده می کند:

```
refresh_pattern -i ^http:\/\/www\.google\.com\/$ 0 20% 360 override-
expire override-lastmod ignore-reload ignore-no-cache ignore-no-store
reload-into-ims ignore-must-revalidate
```

چگونگی برخورد با مسئله انتقال نا تمام درخوساتها

هنگامی که یک کاربر درخواستی را که شامل حجم مشخصی از داده است به سوی اسکوئید می فرستد و پس از مدتی از ادامه ی دریافت آن منصرف می شود و این عمل توسط چندین کاربر تکرار شود در هر مرتبه از ارسال درخواست اسکوئید تلاش می کند این درخواستهای ناتمام را پاسخ دهد. این عمل موجب هدر رفتن مقدار زیادی از منابع سرور مانند حافظه ی رم و واردآمدن بار پردازشی بیهوده ای بر آن می شود .این مشکل از طریق سه دستور quick_abort_min (KB), quick_abort_max (KB) و quick_abort_pct (percent) قابل حل است .برای تمامی درخواستهای لغو شده و به عبارتی نیمه تمام اسکوئید هر سه دستور را چک می کند و بهترین گزینه را انتخاب می نماید مشخصات و نحوه عملکرد این دستورات به شرح زیر است:

- در صورتیکه مقدار داده باقی مانده جهت پاسخ گویی به یک درخواست ناتمام کمتر از مقدار تعیین شده در دستور quick_abort_min باشد ، اسکوئید درخواست مورد نظر را به طور کامل پاسخ می دهد.
- اگر مقدار داده ای که می بایست مبادله شود بیشتر از مقدار تعیین شده در دستور quick_abort_pct (percent) باشد ، اسکوئید سریعاً ادامه ی تبادل داده را متوقف می کند.
- اگر حجم داده های که با کاربر مبادله شده در کل بیش از مقدار درصدی تعیین شده در دستور

quick_abort_pct باشد اسکوئید داده مورد نظر را برای ادامه ی دریافت از سوی کاربر در حافظه نگه داری می کند.

مقادیر دو دستور quick_abort_min و quick_abort_max بر حسب کیلو بایت و یا هر واحد قابل قبول برای تعیین حافظه تعیین می شودو واحد اندازه گیری quick_abort_pct به صورت در صد تعیین می شود.در صورتکیه بخواهید تمامی درخواستهای ناتمام را خاتمه دهید و از ادامه آنها توسط کش سرور جلوگیری کنید باید مقادیر quick_abort_min و quick_abort_max را برابر صفر قرار دهید این حالت تنها زمانی مفید است که پهنای باند موجود محدود باشد.در صورتیکه پهنای باند زیادی در اختیار داشته باشید می توان مقادیر این دستورات را افزایش داد مثال زیر نمونه ای از پیکربندی این سه دستور برای سرورهای با پهنای باند زیاد را نشان می دهد:

```
quick_abort_min 1024 KB
quick_abort_max 2048 KB
quick_abort_pct 90
```

کش کردن درخواستهای نا موفق

هنگامی که کاربران درخواستی را به سوی یک سرور خاص می فرستند در صورتیکه درخواست مورد نظر برروی سرور موجود نباشد با خطای 404 مواجه می شوند همچنین برخی دیگر از درخواستها به دلیل ناکافی بودن مجوز کاربران در دریافت با خطای 403 به معنی عدم اجازه ی مشاهده و یا دریافت داده مواجه می شوند این گونه در خواستهای ناموفق گاها درصد قابل توجهی از کل درخواستهای کاربران را تشکیل می دهد .این درخواستها توسط اسکوئید قابل کش شدن هستند اما قطعا این کار بیهوده ای است و ناشی از عملکرد نادرست وب سرورهای مقصد می باشد به طوریکه در برخی موارد وب سرور مورد نظر به دلیل عدم ارسال هدر Expires در پاسخ های خود موجب می شود اسکوئید این داده ها را در مخزن خود ذخیره نماید. برای حل این مشکل اسکوئید دستور negative_ttl را فراهم کرده است این دستور اسکوئید را مجاب می که این درخواستها را برای مدت زمان مشخصی که در این دستور تعیین می شود در حافظه نگهداری کند.ساختار کلی این دستور به صورت negative_ttl TIME_UNITS می باشد.در نسخه های قبلی اسکوئید مقدار پیش فرض این مقدار 5 دقیقه تعیین شده بود اما در نسخه جدید تر این مقدار به طور پیش گزیده صفر تعیین شده است.

مدیریت هدر های HTTP

تقریبا تمامی درخواستها و داده هایی که به سوی اسکوئید فرستاده می شوند حاوی هدر هایHTTP هستند که اسکوئید قادر به اضافه کردن ،حذف و یا بررسی این هدر ها است.این عمل به طور معمول به منظور حفظ گم نامی کاربران و مخفی ماندن اطلاعات کاربری آنان صورت می گیرد.اسکوئیدازسه دستور request_header_access, reply_header_access,

header_replace به منظور مدیریت این هدر ها استفاده می کند که در ادامه به بررسی آنها می پردازیم.

نکته: در به کارگیری این دستورات نهایت دقت را داشته باشید زیرا این دستورها موجب نقض برخی استاندارد های پروتکل **HTTP** *می شوند.*

مدیریت هدر های HTTP هنگام دریافت درخواستها

دستور request_header_access به همراه ترکیبی از ACL ها به منظور تعیین اینکه کدام یک از هدر های HTTP مربوط به گروه خاصی از کاربران حفظ شود و یا این که هدر ها قبل از پاسخگویی نادیده گرفته شوند .مزیت این دستور این است که به دلیل ترکیب شدن با ACL ها از انعطاف بسیار بالایی برخوردار است.به طوریکه می توان تنها یک کاربر و یا گروه مشخصی از کاربران را تعیین کرده و مانع از ثبت شدن سر فایلهای آنها در فایلهای ثبت رویداد اسکوئید شد ساختار کلی این دستور به صورت زیر است:

```
request_header_access HEADER_NAME allow|deny [!]ACL_NAME ...
```

به طور مثال درصورتکیه بخواهیم فعالیتهای کاربران در شبکه از جمله نوع مرورگر مورداستفاده توسط آنان برای مشاهده صفحات وب مخفی بماند ،با استفاده از کد زیر می توان هدر user-agent را برای تمامی درخواستها مسدود کرده و مانع از ثبت اطلاعات موجود در این هدر در فایلهای ثبت رویداد اسکوئید شد:

```
request_header_access User-Agent deny all
```

در اینجا پارامتر all به نمایندگی از تمامی هدر های user-agent در نظر گرفته می شود . به طور مشابه در صورتیکه مایل باشیم فعالیت های کاربران حتی از دید وب سرورهای مقصد هم مخفی بماند دستور Referer این عمل را به صورت request_header_access Referer deny all انجام می دهد.

مدیریت هدر های HTTP هنگام ارسال پاسخها

مشابه دستور request_header_access دستور reply_header_access هم وجود دارد و وظیفه ی آن حذف و یا به عبارتی مسدود کردن هدر های http هنگام ارسال پاسخ درخواست به سوی کاربر است .نحوه به کارگیری این دستور نیز مانند مورد قبل است به طور مثال برای حذف هدر server به صورت زیر عمل می کنیم:

```
Reply_header_access server deny all
```

به طور پیش فرض تمامی هدر های دریافتی توسط پراکسی سرور در دریافت و ارسال داده ها نگهداری می شود و چیزی

حذف نخواهد شد .

تغییر محتوای موجود در هدر های HTTP

دو دستوری که در بالا معرفی شدند فقط می توانند مانع از ثبت شدن و نمایان شدن هدر ها شوند و امکان هر گونه تغییر در این داده ها از حیطه ی کاری آنها خارج است. تغییر و جابجا کردن محتوای هدر ها از طریق دستورheader_replace امکان پذیر است این دستور پیغام خطای مبنی بر اشتباه بودن این هدر ها را ارسال کرده و محتوای اصلی را با محتوای مورد نظر جابجا می کند. توجه به این نکته ضروری است که این دستور فقط می تواند برروی هدر های درخواستهایی که از طریق دستور request_header_access حذف و یا نادیده گرفته شده اند تغییرات انجام دهد و برروی پاسخها هیچگونه تغییری را اعمال نخواهد کرد. ساختار کلی دستور به صورت header_replace HEADER_NAME TEXT_VALUE است.به طور مثال در کد زیر هدر user-agent را که اطلاعات مربوط به نوع مرورگر کاربران را نگهداری می کند را با اطلاعات زیر جایگزین کرده که این عمل موجب می شود وب سرورهای خارجی احساس کنند تمامی کاربران شبکه ی شما از مرورگر موزیلا فاریرفاکس استفاده می کنند:

```
header_replace User-Agent Mozilla/5.0 (X11; U; Linux i686; en-US;
rv:0.9.3) Gecko/20010801
```

در استفاده از این دستور نیز مانند دو مورد قبلی که در واقع ایجاد تغییرات برروی استاندارد های وب است نهایت دقت را داشته باشید ،هر گونه تغییرات نابجا ممکن است باعث مشکلات ناخواسته در نمایش وب سایتها و همچنین محتوای آنها خواهد داشت.

پیکربندی DNS سرور

DNS سرورها وظیفه ی ترجمه ی آدرس وب سایتهای مختلف به آدرس IP آنها را برعهده دارند.به ازای هر درخواستی که از طرف کاربران به سوی اسکوئید ارسال می شود اولین گام در پاسخگویی به این درخواستها ترجمه ی آدرس URL به آدرس IP آن است در این میان هرچقدر ترجمه آدرس با سرعت بیشتری انجام شود ، سرعت پاسخگویی و در نتیجه سرعت ارسال درخواست به کاربر بیشتر می شود.اسکوئید به منظور تسریع در انجام این کار از یک سیستم DNS Cache داخلی استفاده می کند که در آن آدرسهای قبلی را به همراه آدرس ip آنها نگهداری می کند و در موارد بعدی به جای مراجعه به DNS سرورهای اصلی از آنها برای پاسخگویی به درخواستها استفاده می کند . برای غیر فعال کردن آن هنگام کامپایل دستور $ **configure --disable-internal-dns/.** را همراه سایر گزینه های پیکربندی به کار ببرید. یکی دیگرگزینه های موجود استفاده از یک سرویس DNS برروی همان سرور است که در ادامه به چگونگی پیکربندی هر کدام می پردازیم.

مشخص کردن مسیر سرویس دهنده DNS

دستور cache_dns_program به منظور مشخص کردن مسیر سرویس دهنده DNS خارجی به کار می رود برای تعیین مسیر مورد نظر به صورت cache_dns_program /path/to/dnsprogram که در آن به جای قسمت دوم مسیر مورد نظر خود را وارد نمائید.

مشخص کردن DNS Name server

به طور پیش فرض، اسکوئید از DNS های موجود در مسیر /etc/resolv.conf استفاده می کند.در صورتکیه بخواهیم از DNS Server های دیگری استفاده کنیم باید دستور dns_nameservers را به همراه نام DNS سرور مورد نظر خود به صورت زیر کار ببریم :

dns_nameservers 4.2.2.4 4.2.2.2 8.8.8.8

در اینجا آدرس سه مورد از DNS سرورهای سرارسری موجود را وارد تعیین کرده ایم .این دستور موجب می شود تا اسکوئید فقط از این DNS سرورها جهت ترجمه ی آدرس سایتها استفاده کند.

مشخص کردن مسیر فایل میزبان

علاوه بر DNS سرورها ، اسکوئید می تواند نام های میزبان را همراه با آدرسip آنها در یک شبکه ی محلی شناسایی کند. مسیر نگهداری نامهای میزبان در /etc/hosts قرار دارد و معمولا شامل نام سرورهای موجود در شبکه محلی موجود در شبکه است. دستور hosts_file برای تعیین مسیر این فایل به کار می رود وساختار دستوری آن به صورت hosts_file /etc/hosts می باشد.در صورتکیه بخواهیم اسکوئید به این نام هادسترسی نداشته باشد میتوان مقدار این دستور را none تغییر تعیین کرد.

تعیین میزان زمان تاخیر در در خواست های DNS

در صورتکیه DNS سرورهای مورد استفاده ی اسکوئید نتوانند در یک زمان مشخص که مقدار آن دستور dns_timeout تعیین می شود به درخواستهای ارسال شده به آنها پاسخ دهند اسکوئید آن ها را خارج از دسترس فرض کرده و DNS سرورها های بعدی را بررسی می کند مقدار پیش فرض این دستور دو دقیقه است که با توجه به افزایش سرعت شبکه های اینترنتی و بالطبع بالا رفتن انتظارات کاربران این مقدار زیاد به نظر می رسد و می توان این مقدار را به یک دقیقه و یا حتی کمتر کاهش

داد: dns_timeout 50 seconds

کش کردن پاسخهای DNS سرورها

آدرس های ip مربوط به بسیاری از آدرس ها و دومین های موجود در شبکه به ندرت تغییر می کنند و معمولا به صورت یک مقدار ثابت باقی می مانند بنابراین ذخیره شدن این آدرسها به همراه آدرس IP آنها در سرور می تواند موجب کاهش زمان ترجمه و درنتیجه افزایش سرعت پاسخگویی می شود .برای ذخیره سازی این نامها دو دستور positive_dns_ttl و negative_dns_ttl پیش بینی شده است .دستور positive_dns_ttl حداکثر زمانی را که یک DNS سالم می تواند کش شود را مشخص می کند و negative_dns_ttl هم کمترین زمانی را که یک DNS غیر فعال کش می شود را تعیین می کند به مثال زیر توجه کنید.

```
positive_dns_ttl 8 hours
negative_dns_ttl 30 seconds
```

در این مثال مقادیر 8 ساعت و 30 ثانیه به ترتیب به عنوان حداکثر و حداقل این زمانها در نظر گرفته شده اند.

تعیین حجم DNS های کش شده

اسکوئید با استفاده از DNS سرورها قادر به ترجمه ی آدرس های دومین به IP آنها و برعکس تبدیل یک آدرس IP به آدرس دومین مربوط به درخواستهایی که Acl های نوع dstdomain را شامل می شوند می باشد.این جستجو و تبدیل آدرس ها جهت استفاده ی دوباره ذخیره می شوند و اسکوئید به منظور کنترل هر چه بیشتر بر این فرایند دستورات ipcache_size (number), ipcache_low (percent), ipcache_high (percent), (fqdncache_size (number را فراهم آورده است. ipcache_size تعداد آدرسهای دومین ورودی را که به آدرس ip مربوط به خود ترجمه شده اند را جهت کش شدن تعیین می کند یکی از ویژگیهای این آدرسها استفاده ی بسیاری کم آنها از فضای حافظه است به طوریکه میتوان هرازان عدد از این آدرسها را در یک فضای کوچک از حافظه ی اصلی نگهداری کرد.مقدار پیش فرض این دستور 1024 تعیین شده است که میتوان این مقدار را در کش سرورهای با ترفیک بالا به مقادیر بسیار بزرگتر مثلا 15000 افزایش داد.دستورات ipcache_low و ipcache_high مقادیر ورودی این آدرس ها را کنترل می کنند و از ورود بیش از حد مجاز آنها به حافظه ی اصلی جلوگیری می کنند ومقادیر پیش فرض تعیین شده برای این دو دستور به ترتیب 95 و 97 درصد می باشد بدین معنی که همواره تعداد آدرسهای نگهدای شده در حافظه مقداری میان 95 تا 97 درصد از کل تعداد تعیین شده را شامل می شود.دستور fqdncache_size تعداد کلی آدرسهای ترجمه شده جهت ذخیره سازی در حافظه ی اصلی را تعیین می کند.همانند دستورات قبلی این دستور نیز حافظه ی بسیاری کمی را اشغال می کند ،مقدار پیش فرض تعیین شده برای این دستور

1024 آدرس تعیین شده است که با توجه به شرایط کش سرور و میزان ترافیک آن میتوان این مقدار را به بیش از 10000 افزایش داد.

فایلهای ثبت رویداد

سرویس دهنده اسکوئید تمامی درخواستهای کاربران و همچنین تمامی رویدادهای مربوط به اسکوئید مانند خطاهای احتمالی را در فایلهای ثبت رویداد به منظور شناسایی مشکلات ذخیره می کند .در مورد ساختار و جزئیات این فایلها در فصل پنجم به طور مفصل بحث خواهیم کرد اما در ادامه به صورت مختصر به بررسی آنها می پردازیم.

پشتیبان گیری و گرد سازی فایلهای ثبت رویداد

هنگامی که اسکوئید برای مدتی در شبکه به فعالیت می پردازد ، به دلیل ثبت تمامی درخواستها و خطاهای احتمالی حجم فایلهای ثبت رویداد به تدریج بزرگ می شوند و مقدار زیادی از فضای هارد دیسک را اشغال می کنند راهکار اسکوئید برای جلوگیری از پرشدن فضای آزاد دیسک سخت ، گرد سازی این فایلها و نگهداری آنها به عنوان پشتیبان است اسکوئید به طور پیش فرض 10 نسخه از فایلهای گرد شده را به عنوان پشتیبان به منظور مراجعه به آنها در مواقع ضروری نگهداری می کند،این مقدار مانند بسیاری از دستورات دیگر قابل تغییر است .دستور logfile_rotate بدین منظور ایجاد شده است و نحوه کارکرد این دستور به صورت logfile_rotate n value است که به جای n value یک عدد مثلا 5 قرار می گیرد و معنای آن این است که اسکوئید 5 نسخه پشتیبان از فایلهای ثبت رویداد گرد شده را در بازه زمانی متفاوت نگهداری می کند توجه داشته باشید این دستور یکی از دستورات بسیار مهم به منظور مدیریت فضای دیسک است و حتما از یک مقدار متناسب با فضای آزاد کل دیسک سخت برای این دستور در نظر بگیرید.مثال زیر مقدار 20 را برای این دستور در نظر گرفته است:

```
logfile_rotate 20
```

سفارشی سازی ثبت رویدادها

یکی از وظایف مهم پراکسی سرورها جمع اوری اطلاعات از کارکرد شبکه و خصوصا فعالیت کاربران شبکه است .همچنین تهیه ی گزارشی کامل از اطلاعات مبادله شده میان پراکسی سرور و کاربران از مهمترین وظایف اسکوئید به شمار می رود بدین منظور اسکوئید تمامی درخواستهای کاربران را در یک فایل به نام access.log ذخیره می کند.درصورتکیه بخواهیم درخواستهای یک کاربر مشخص و یا گروهی از کاربران ثبت نشود می توان از دستور log_access همراه با acl ها به صورت زیر عمل کرد:

```
acl ceo_laptop src 192.0.2.21
log_access deny ceo_laptop
```

در این مثال تمامی درخواستهایی که از آدرس 192.0.2.21 به سوی اسکوئید فرستاده می شود ثبت نخواهد شد.

نکته: به کارگیری این دستور در مقیاس وسیع موجب کاهش اعتبار گزارشات مربوط به عملکرد و بازده اسکوئید خواهد شد.

Buffered logs

تمامی درخواستهای کاربران به طور مستقیم و بدون هیچ واسطه ای در فایهای ثبت رویداد ذخیره می شود .استفاده از فضای بافر موجب بهبود عملکرد ثبت رویدادها در شرایط سخت کاری می شود البته به دلیل تغییرات صورت گرفته در نسخه های به روز شده ی اسکوئید استفاده از این دستور کاربرد چندانی ندارد و ما هم از توضیح بیشتر خودداری می کنیم!

Strip query terms

کلمات کلیدی موجود در درخواستهای HTTP معمولا شامل اطلاعات شخصی کاربران که افشای آنها گاها موجب نقض حریم شخصی آنان می شود.به طور پیش فرض اسکوئید تمامی اطلاعات جستجو شده را در آدرس url مورد جستجو به طور دقیق نگهداری می کند اما یکی از اشکالات فرایند طولانی شدن بیش از حد آدرسهای ثبت شده و سخت شدن شدن خواندن و شناسایی آنها هنگام بروز مشکل و در نتیجه طولانی شدن زمان اشکال زدایی اسکوئید است با دستور strip_query_terms off می توان این قابلیت را به طور دائم و یا موقت غیر فعال نمود بدین ترتیب هردرخواستی که از سوی کاربران دریافت می شود ،به جای ثبت آدرس دقیق داده مورد نظر فقط قسمت اول نشانی آن مثلا نام وب سایت جستجو شده ذخیره شده و سایر جزئیات دقیق آن مثلا آدرس دقیق داده برروی سرور نادیده گرفته می شود.

تغییر نشانی و بازنویسی آدرس های URL

url rewtite ها و redirector ها از جمله ابزارهای جالبی هستند که اسکوئید از آنها برای تغییر نشانی آدرسهای وب از مسیر اولیه و هدایت آنها به سوی یک مسیر تعیین شده به کار می روند به طور مثال میخواهیم آدرس یک وب سایت و یا یک محتوای وب را با یک آدرس دلخواه جایگزین کرده و زمانی که کاربر آدرس مورد نظر را در مرورگر خود وارد کند آدرس دیگری که ما تعیین کرده ایم به جای آدرس اصلی نمایش داده شود.url rewrite ها به صورت نرم افزارهای جداگانه ای در کنار اسکوئید استفاده می شوند و کسانی که با زبانهای برنامه نویسی perl و python آشنایی دارند می توانند کدهای نوشته شده ی خود را همراه با اسکوئید به کار ببرند.به طور معمول با استفاده از ابزارهای استاندارد اسکوئید مانند acl ها , http_access ها می توان تعداد محدودی از آدرسها را بازنویسی و یا منتقل نمود مثلا می توان یک آدرس مشخص را به صفحه ی خطاهای اسکوئید هدایت کرد و یا آن را فیلتر نمود.در فصل های آینده به طور مفصل در این مورد صحبت خواهد شد.

تعیین نام کاربری اسکوئید

همانطور که پیش تر هم گفته شد اسکوئید نمی تواند تحت مجوز نام کاربری ریشه یا root که بالاترین حساب کاربری در تمامی سیستم های یونیکسی و لینوکسی است فعالیت کند و دلیل آن هم این است که حساب کاربری root بالاترین سطح مجوز و اختیارات را در تمامی این سیستم ها در اختیار دارد به همین دلیل هنگامی که اسکوئید راه اندازی می شود حساب کاربری خود را از root به یک حساب کاربری از پیش تعیین شده توسط دستور cache_effective_user منتقل می کند .به طور پیش فرض نام کاربری nobody برای این دستور در نظر گرفته شده است که ما به وسیله دستور بالا می توانیم نامهای کاربری جدید تعریف کنیم البته در برخی از سیستم عامل ها نام پیش فرض همان squid تعیین شده است. به مثال زیر دقت کنید:

```
cache_effective_user  squid
```

در اینجا squid به عنوان نام کاربری اسکوئید و یا همان user ID تعیین شده است.توجه داشته باشید در صورتیکه برای تغییر نام پیش فرض کاربری دلیل خاصی ندارید برای جلوگیر یاز بروز برخی مشکلات ناخواسته از تغییر آن خودداری کنید.

پیکربندی نامهای میزبان بر ای پراکسی سرور

اسکوئید از نامهای میزبان به منظور برقراری ارتباط و ارسال درخواستهای کاربران به سایر کش سرورهای همسایه استفاده می کند.برای تعیین نام میزبان دو دستور visible_hostname و unique_hostname درنظر گرفته شده که برای مشخص کردن نام میزان برای اهداف مختلف استفاده می شود .

نام میزبان قابل مشاهده برای همه

نام میزبانی که توسط دستور visible_hostname تعیین می شود در تمامی صفحات خطای اسکوئید هنگام بروز خطا و مشکلات شبکه به کاربران نمایش داده می شودکه معمولا در پایان صفحات خطا و در قسمت امضا نمایان می شود.

```
visible_hostname  proxy.example.com
```

نام میزبان منحصربه فرد برای سرور

در صورتکیه بخواهیم برای تمامی پراکسی سرورهای موجود در شبکه یک نام میزبان یکسان مثلا proxy.example.com تعیین شود می توان از دستور visible_hostname استفاده نمود. در صورت استفاده از یک نام میزان یکسان برای تمامی سرورهای پراکسی موجود در شبکه موجب ایجاد مشکلات ارتباطی هنگام ارجاع درخواستهای کاربران خواهد شد .برای

جلوگیری از این مشکل با استفاده از دستور unique_hostname می توان یک نام میزبان منحصر به فرد برای پراکسی سرور به صورت زیردر نظر گرفت:

unique_hostname Squid-Proxy

کنترل بیشتر بر فرایند ارجاع درخواستها

در صورتکیه در شبکه بیش از یک کش سرور وجود داشته باشد اسکوئید تلاش می کند تا با آنها ارتباط برقرار کرده و درخواستهای ذخیره شده و سایر منابع خود را با آنها به اشتراک بگذارد. به منظور کنترل هرچه بیشتر بر این فرآیند و سفارشی سازی بیشتر آن دستورات مختلفی مانند , always_direct, never_direct ,hierarchy_stoplist prefer_direct,cache_peer_access ایجاد شده اند در ادامه به معرفی این دستورات می پردازیم.

Always direct

در صورتیکه بخواهیم درخواستهای ارسال شده به سوی اسکوئید به طور مستقیم از طریق سرورهای اصلی و درخواستها برای پاسخگویی احتمالی به پراکسی سرورهای همسایه ارجاع داده نشوند از دستور always_direct استفاده می شود.نحوه به کارگیری این دستور بسیار شبیه بوده http_access و نحوه به کارگیری آن به صورت زیر است:

always_direct allow|deny [!]ACL_NAME

این دستور درمواردی که کاربران در یک شبکه داخلی هستند بسیار مفید است زیرا با وجود یک شبکه داخلی LAN با سرعت مناسب نیازی به ارسال درخواست به سرورهای همسایه احساس نمی شود.

acl lan_servers dst 192.0.2.0/24

always_direct allow lan_servers

این دستورات موجب می شوند درخواستهایی که از محدوده ی آدرس تعیین شده در lan_servers به سوی اسکوئید ارسال میشوند به طور مستقیم به سرورهای مورد نظر مقصد ارجاع داده شوند هیچ یک از این درخواستها به سوی سرورهای همسایه فرستاده نشوند.

Never direct

کارکرد این دستور مخالف دستور always direct میباشد.در به کارگیری این دستور باید دقت داشته باشید،درصورتیکه می خواهید درخواستهای تمامی کاربران و حتی کاربران شبکه محلی به پراکسی سرورهای همسایه منتقل شوند از این دستور به صورت never_direct allow all استفاده کنید.این کار عاقلانه به نظر نمی رسد و نتیجه ی آن ایجاد تاخیر برای کاربران

شبکه محلی است اما با استفاده از دستورات acl می توان برای گروهی از کاربران موجود در شبکه ی محلی استثنا قائل شده و درخواستهایی را که از طریق سرورهای محلی قابل پاسخگویی هستند را از طریق همان سرورها پاسخ داده شوند :

```
acl  lan_servers dst 192.0.2.0/24
never_direct deny lan_servers
never_direct allow all
```

این دستورات تمامی درخواستها را به جز آنهایی که در محدوده ی آدرس تعیین شده در lan_servers هستند را به سایر پراکسی سرورها ارجاع می دهد.

Hierarchy stoplist

این دستور یکی از ساده ترین دستوراتی است که توسط آن می توان از ارجاع درخواستهایی که حاوی یکی از محتوای تعیین شده در این دستور هستند به سوی کش سرورهای همسایه جلوگیری کرد. شکل کلی دستور به صورت hierarchy_stoplist word1 word2 word3 ... است که به جای هرکدام از کلمات word نام یک فرمت یک داده مشخص قرار می گیرد.به کارگیری این دستور مشکلات ناشی از نمایش کدهای امنیتی موجود در برخی وب سایتهای پویا را به حداقل کاهش می دهد.شکل کلی به کارگیری ایندستور به صورت زیر است:

```
hierarchy_stoplist cgi-bin jsp ?
```

در این مثال ،این دستور مانع از ارسال درخواستهایی که حاوی محتوای ؟ , cgi-bin , jsp هستند به سمت سرورهای همسایه می شود.

نکته: در صورت استفاده از دستور **never_direct** *این دستور بی اثر می شود بنابراین در یک فایل پیکربندی فقط می توان یکی از این دو دستور را به کارگرفت.*

TCP outgoing address

این دستور برای هدایت ترافیک خروجی کش سرور به طرف یک اینترفیس مشخص به کار می رود به عبارت دیگر درصورتیکه بخواهیم ترافیک خروجی از کش سرور برای تعداد مشخصی از کاربران از یک اینترفیس و آدرس ip جداگانه صورت گیرد از این دستور استفاده می کنیم شکل کلی این دستور به صورت tcp_outgoing_address ip_address [[!]ACL_NAME]است که در این کد به جای عبارت ip_address آدرس ip اینترفیس خروجی کش سرور تعیین می شود و قسمت Acl هم کاملا اختیاری می باشد در مثال زیر چگونگی عملکرد این دستور مشخص شده است:

```
acl special_network src 192.0.2.0/24
tcp_outgoing_address 198.51.100.25 special_network
tcp_outgoing_address 198.51.100.86
```

در این دستورات ، آدرس اینترفیس خارجی کش سرور برای کاربران موجود در محدوده ی آدرس 192.0.2.0/24 آدرس 198.51.100.25 تعیین و برای سایر درخواستها آدرس خروجی 198.51.100.86 تعیین شده است.به طور کلی این دستور به منظور هدایت ترافیک قسمتی از شبکه به سوی یک آدرس مشخص به کار می رود.

PID filename

اسکوئید نیز دقیقا مانند سایر نرم افزارهای لینوکس و یونیکس شماره ID خود را در یک فایل متنی به نام squid.pid نگهداری می کند و در واقع سیستم عامل از طریق این کد اسکوئید را شناسایی کرده و با آن ارتباط برقرار می کند. pid_filename /var/run/squid.pid مسیر ایجاد این فایل را تعیین می کند و برای غیر فعال کردن آن می توان از دستور pid_filename none استفاده کرد. غیر فعال کردن این دستور به هیچ عنوان توصیه نمی شود زیرا درصورت غیر فعال شدن آن سیستم عامل قادر به فعال سازی و راه اندازی مجدد اسکوئید نخواهد بود همچنین فرایند گرد سازی فایلهای ثبت رویداد (logfile rotation) نیز غیر ممکن خواهد شد.

Client netmask

به طور پیش فرض اسکوئید آدرس ip کاربران را برای هر درخواست به طور کامل شناسایی و ذخیره می کند که این امر در برخی مواقع باعث نقض حریم شخصی افراد می شود. به منظور احترام به حریم شخصی کاربران می توان آدرس ip حقیقی آن ها را پنهان کرد .به مقال زیر توجه کنید:

```
client_netmask 255.255.255.0
```

در صورتیکه آدرس ip یک کاربر در این شبکه 192.0.2.21 باشد آدرس 192.0.2.0 به جای 192.0.2.21 در فایل ثبت رویداد نگهداری می شودو ۸ بیت آخر آدرس ها را با صفر جایگزین می کند.

خلاصه فصل

در این فصل مطالب متنوعی را در مورد چگونگی پیکربندی اسکوئید و سفارشی سازی آن مطابق با وضعیت شبکه و نیازهای خود آموختیم همچنین با نحوه دستورات پیکربندی اسکوئید و چگونگی ترکیب آنها با یکدیگر به به کارگیری آنها در دستورات متفاوت به منظور افزایش کارایی و امنیت کش سرور خود مطالبی را آموختیم.

به طور کلی موضوعات زیر در این فصل شرح داده شد:

- ساختار دستورات پیکربندی اسکوئید و همچنین انواع دستورات پیکربندی و مقادیر پیش فرض و قابل استفاده در آنها مانند دستورات تک مقدار و یا چند مقدار.
- کش کردن اطلاعات برروی RAM و دیسک سخت و چگونگی انجام آن همچنین بررسی مزایا و معایب هر یک و انتخاب بهترین روش به منظور دستیابی به بالاترین کارایی ممکن.
- بهینه سازی فرایند کش کردن و ذخیره سازی اطلاعات با استفاده از کنترل بیشتر بر روی هدر های HTTP.
- نحوه استفاده از آدرسهای DNS در اسکوئید و مشخص کردن DNS سرور های مورد استفاده در اسکوئید و کش کردن آنها با هدف کاهش زمان ترجمه ی آنها .

در این فصل همچنین با نحوه کارکرد دستورات acl و http_access و نحوه ترکیب و به کارگیری آنها به مظنور اعمال کنترل و محدودیت در دسترسی کاربران آشنا شدیم.در فصل بعد با نحوه راه اندازی اسکوئید و دستورات مرتبط با آن بیشتر آشنا خواهیم شد.

فصل سوم

راه اندازی اسکوئید

در فصل گذشته موضوعات مختلفی در مورد نصب ،کامپایل و پیکربندی اسکوئید را آموختیم و توانستیم اسکوئید را از طریق روش های مختلف برروی یک سیستم عامل لینوکس نصب کنیم.در این فصل مطالب مفصلی در مورد چگونگی راه اندازی و اجرای اسکوئید،دستورات مختلف کنترل و مدیریت آن و همچنین چگونگی رفع اشکال آن خواهیم آموخت.

به طور کلی مطالب مورد بحث در این فصل شامل موارد زیر است:

- دستورات مختلف خط فرمان جهت راه اندازی اسکوئید
- تحلیل تنظیمات پیکربندی اسکوئید به منظور رفع اشکال نحوی آن
- استفاده از فایلهای پیکربندی متناوب و مختلف جهت بررسی کارکرد اسکوئید
- راههای مختلف راه اندازی اسکوئید
- گرد سازی فایلهای ایجاد شده توسط اسکوئید

دستورات خط فرمان

سرویس دهنده اسکوئید به دلیل پیچیدگی و گستردگی فایل پیکربندی و همچنین تنظیمات بسیار متنوع ، از دستورات خط فرمان به منظور مدیریت و کنترل این تنظیمات بهره می گیرد. به طور کلی هر تغییری که می بایست در اسکوئید صورت گیرد منشا آن فایل پیکربندی اصلی آن یعنی squid.conf است.دستورات مورد استفاده در اسکوئید مانند سایر دستورات موجود در سیستم عاملهای یونیکس و لینوکس از قسمتهای پیشوندی و پسوندی تشکیل شده اند که دستورات پیشوندی در واقع ساختار کلی دستور، و مسیر آن را نشان می دهد و قسمت های پسوندی نیز گزینه های قابل استفاده و به اصطلاح option های هر دستور را نشان می دهد .گرچه دستورات بسیاری زیادی برای اسکوئید در نظر گفته شده اما تعداد زیادی از آنها به منظور رفع اشکال و خطایابی فایل پیکربندی به کار می روند و موارد استفاده کمی دارند و تعدادی از آنها برای مواردی چون متوقف کردن و یا راه اندازی مجدد اسکوئید مورد استفاده قرار می گیرند.همانطور که در فصل اول آموختیم ،مسیر فایهای اجرایی اسکوئید بسته به مسیر تعیین شده در عبارت prefix-- همراه با کمپایل اسوئید متفاوت است اما در صورتکیه مسیر فایلهای نصب اسکوئید هنگام کامپایل تغییر نیافته باشند ،مسیر یش فرض این دایرکتوریها در مسیر usr/local/squid/sbin/squid/ خواهد بود اما در صورت هرگونه تغییر این مسیر به صورت sbin/squid {prefix}تعیین می شود که در قسمت {prefix} باید مسیر تعیین شده هنگام کامپایل قرار گیرد به هر حال برای راه اندازی اسکوئید باید یکی از این دو روش را به کار بگیریم:

- در صورتکیه پیش فرض هنگام کامپایل اسکوئید تعیین شده باشد ،هنگام راه اندازی اسکوئید از دستور usr/local/squid/sbin/squid/ که فایل اجرایی اسکوئید در آنجا قرار دارد استفاده می کنیم.
- در صورتکیه مسیری متفاوت از مسیر پیش فرض تعیین شده باشد باید از دستور {prefix}/usr/local/squid/sbin/squid به منظور راه اندازی آن استفاده کنیم.

برای بسیاری از افرادی که با خط فرمان کار می کنند ،تایپ تمامی این مسیر ممکن است کاری خسته کننده به نظر آید ،برای حل این مسئله می توان از یکی از توانایی های خوب سیستم عاملهای لینوکس و یونیکس استفاده کرد،و آن هم استفاده از دستور export و متغیر PATH مربوط به پوسته ی لینوکس است به نظور کوتاه کردن خطوط راه اندازی اسکوئیداست .به مثال زیر توجه کنید:

```
$ export PATH=$PATH:/usr/local/squid/sbin/
```

به طور مشابه می توان فرمان زیر را به کار برد:

```
$ export PATH=$PATH:/opt/squid/sbin/
```

می توانیم هر کدام از دستورات بالا را در فایل Bashrc /.~یا bash_profile/.~ کپی کرده تا هنگام باز کردن ترمینال جدید دستورات ایجاد شده به طور پیاپی اجرا نشوند.

پس از انجام تنظیمات بالا ،شما میتوانید اسکوئید را به سادگی هرچه تمام تر به صورت زیر اجرا کنید:

```
$ squid
```

این دستور همواره اسکوئید را پس از فراخوانی تنظیمات موجوددر فایل پیکربندی squid.conf راه اندازی می کند.

نکته: همواره قبل از به کارگیری این دستورات ، از مسیر اصلی نصب اسکوئید هنگام کامپایل آن اطمینان حاصل نمائید همچنین به منظور رعایت استاندارد ها موجود در این سیستم عاملها همواره مسیر نصب اسکوئید را مانند سایر سرویس دهنده های لینوکس تعیین کنید.

مشاهده فهرست دستورات موجود

قبل از انجام هر کاری باید با دستورات موجود در اسکوئید و موارد استفاده از آنها آشنا شویم.برای مشاهده لیست فرمانهای قابل اجرا دستور usr/local/squid/sbin/squid -h/ و در صورتکیه اسکوئید را از طریق بسته های آماده نصب کرده اید دستور squid -h را وارد کنید خروجی مشابه زیر خواهد بود.

```
Usage: squid [-cdhvzCFNRVYX] [-s | -l facility] [-f config-file] [-[au]
port] [-k signal]
-a port Specify HTTP port number (default: 3128).
-d level Write debugging to stderr also.
-f file Use given config-file instead of
/opt/squid/etc/squid.conf.
-h Print help message.
-k reconfigure|rotate|shutdown|interrupt|kill|debug|check|parse
Parse configuration file, then send signal to
running copy (except -k parse) and exit.
-s | -l facility
Enable logging to syslog.
```

-u port Specify ICP port number (default: 3130), disable with 0.

-v Print version.

-z Create swap directories.

-C Do not catch fatal signals.

-F Don't serve any requests until store is rebuilt.

-N No daemon mode.

-R Do not set REUSEADDR on port.

-S Double-check swap during rebuild.

همانطور مشاهده می کنید در مقابل هر دستور توضیحی مختصر در مورد نوع و کاربرد آن آمده است که برای کسانی که به تازگی یادگیری اسکوئید را آغاز کرده اند بسیار مفید است .با مراجعه به آدرس http://linux.die.net/man/8/squid اطلاعات بیشتری در این مورد کسب خواهید کرد.

به دست آورن اطلاعاتی در مورد نسخه ی اسکوئید

همانطور که میدانید هنگام نصب و کامپایل اسکوئید می توان برخی از ویژگیهای مورد استفاده در آن را فعال یا غیر فعال کرد. برای آگاهی از اینکه کدام یک از این ویژگیهای فعال و کدام غیر فعال شده اند و همچنین آگاهی از نسخه نصب شده از دستور squid -v استفاده می شود، این دستور اطلاعات کاملی در مورد نسخه ی نصب شده ی اسکوئید به ما می دهد خروجی این دستور مشابه خطوط زیر است:

Squid Cache: Version 3.1.11

Configureoptions:'--build=x86_64-linux-gnu''--prefix=/usr''--includedir=${prefix}/include''--mandir=${prefix}/share/man''--infodir=${prefix}/share/info''--sysconfdir=/etc''--localstatedir=/var''--libexecdir=${prefix}/lib/squid3''--srcdir=.''--disable-maintainer-mode''--disable-dependency-tracking''--disable-silent-rules''--datadir=/usr/share/squid3''--sysconfdir=/etc/squid3' '--mandir=/usr/share/man' '--with-cppunit-basedir=/usr' '--enable-inline''--enable-async-io=8''--enable-storeio=ufs,aufs,diskd''--enable-removal-policies=lru,heap' '--enable-delay-pools' '--enable-cache-digests' '--enable-underscores' '--enable-icap-client''--enable-follow-x-forwarded-for''--enable-

auth=basic,digest,ntlm,negotiate'

این خطوط بسته به نوع نسخه نصب شده و تنظیمات اعمال شده هنگام نصب و کامپایل متفاوت است همانطور که مشاهده می کنید در خط اول شماره ی نسخه و در خطوط بعدی ویژگیهای های فعال شده در این نسخه را نشان می دهد.

ساخت دایرکتوری های کش

در فصل گذشته با ساختار مخزن کش و خصوصیات آن آشنا شدیم ، در صورتکیه قبلا مسیر و نوع این دایرکتوریها را در فایل پیکربندی تعریف کرده اید با دستور squid -z می توان این دایرکتوریها را برروی دیسک سخت ایجاد کرد خروجی دستور مشابه خطوط زیر خواهد بود:

2010/01/18 20:10:50| Creating Swap Directories

2010/01/18 20:10:50| Making directories in /squid_cache/00

2010/01/18 20:10:50| Making directories in /squid_cache/01

2010/01/18 20:10:50| Making directories in /squid_cache/02

این خطوط نشان دهنده موفقیت آمیز بودن عملیات ساخت دایرکتوریهای کش است ،در صورتکیه به دلایلی مانند کمبود مجوز های دسترسی و یا مسیر اشتباه دایرکتوریها خطایی رخ دهد، پیغام خطایی شامل دلایل بروز خطا در خروجی نمایان می شود و شما می بایست پیش از اجرای دوباره ی دستور دلایل خطاهای مشخص شده را برطرف و مجددا دستور squid -z را اجرا کنید.توجه داشته باشید از این دستور می توان در مواردی که فضای اختصاص یافته برای مخزن کش تمام شده باشد برای اضافه کردن دایرکتوریهای جدید هم استفاده کرد اما همواره ابتدا باید مسیر دایرکتوریهای جدید را در squid.conf اضافه کرده و سپس این دستور را اجرا کنید.

تغییر نام فایل پیکربندی اصلی

نام فایل اصلی پیکربندی اسکوئید به صورت پیش فرض squid.conf است که در مسیر پیش فرض etc/squid/quid.conf/ و یا usr/local/squid/etc/squid.conf/ قرار دارد ، اسکوئید این امکان را فراهم آورده است که بتوان نام فایل پیکربندی را نیز تغییر داده و برای آن نامهای متفاوتی در نظر گرفت .مزیت اصلی این کار بررسی و آزمایش تنظیمات پیکربندی مختلف بدون نیاز به تغییر کلیه ی تنظیمات است برای این کار فرمان f- را همراه با squid به

صورت زیر به کار می بریم:

```
squid -f /etc/squid.once.conf
# OR
squid -f /etc/squid.two.conf
```

در صورتکیه اسکوئید به همراه فرمانهای بالا اجرا شود کلیه تنظیمات موجود در squid.conf نادیده گرفته می شود و اسکوئید فایل پیکربندی خود را از روی یکی از دو مسیر بالا دریافت می کند.همانطور که ملاحظه می کنید این دستور تست و رفع اشکال و بررسی تنظیمات کلی را سرعت می بخشد.

مشاهده فرایند بازخوانی فایل پیکربندی هنگام راه اندازی اسکوئید

اگر فرمان squid را به تنهایی در خط فرمان اجرا کنید خروجی این دستور فقط خطاها و هشدارهای احتمالی موجود را نشان می دهد در صورتکیه بخواهیم فرایند راه اندازی و فراخوانی تنظیمات پیکربندی را مشاهده کنیم کافی است دستور squid -d ۲ را در خط فرمان اجرا کنیم،خروجی این دستور مشابه خطوط زیر است:

```
2012/01/14 11:26:28| Starting Squid Cache version 3.1.11 for x86_64-pc-linux-gnu...
2012/01/14 11:26:28| Process ID 8265
2012/01/14 11:26:28| With 1024 file descriptors available
2012/01/14 11:26:28| Initializing IP Cache...
2012/01/14 11:26:28| DNS Socket created at [::], FD 7
2012/01/14 11:26:28| DNS Socket created at 0.0.0.0, FD 8
2012/01/14 11:26:28| Adding nameserver 4.2.2.4 from squid.conf
2012/01/14 11:26:28| Adding nameserver 217.218.127.127 from squid.conf
2012/01/14 11:26:28| Adding nameserver 8.8.8.8 from squid.conf
2012/01/14 11:26:28| Unlinkd pipe opened on FD 13
2012/01/14 11:26:28| Local cache digest enabled; rebuild/rewrite every 3600/3600 sec
2012/01/14 11:26:28| Store logging disabled
```

```
2012/01/14 11:26:28| Swap maxSize 51200000 + 3072000 KB, estimated 4174769 objects
2012/01/14 11:26:28| Target number of buckets: 208738
2012/01/14 11:26:28| Using 262144 Store buckets
2012/01/14 11:26:28| Max Mem  size: 3072000 KB
2012/01/14 11:26:28| Max Swap size: 51200000 KB
2012/01/14 11:26:28| Version 1 of swap file with LFS support detected...
2012/01/14 11:26:28| Rebuilding storage in /var/spool/squid3/cache (CLEAN)
2012/01/14 11:26:28| Using Least Load store dir selection
2012/01/14 11:26:28| chdir: var/spool/squid3: (13) Permission denied
2012/01/14 11:26:28| Current Directory is /root
2012/01/14 11:26:28| Loaded Icons.
2012/01/14 11:26:28| Accepting  intercepted HTTP connections at 0.0.0.0:3128, FD 16.
2012/01/14 11:26:28| Accepting  HTTP connections at [::]:8080, FD 17.
2012/01/14 11:26:28| Accepting ICP messages at [::]:3130, FD 18.
2012/01/14 11:26:28| HTCP Disabled.
2012/01/14 11:26:28| Accepting SNMP messages on [::]:3401, FD 19.
2012/01/14 11:26:28| Squid plugin modules loaded: 0
2012/01/14 11:26:28| Adaptation support is off.
2012/01/14 11:26:28| Ready to serve requests.
...
```

در این دستور پس از فرمان -d عدد ۲ قرار گرفته است در صورتکیه به جای عدد ۲ ، صفر را وارد کنیم خروجی آن فقط خطاهای بسیار جدی و حیاتی نمایش داده می شود ، عدد ۱ خطاها و هشدارهای مهم را نمایش می دهد و بالاخره عدد ۲ و یا بالاتر از ۲ کاملترین خروجی را که شامل کلیه ی مراحل راه اندازی اسکوئید می باشد را نمایش می دهد. این دستور یکی از بهترین فرمانهای رفع اشکال اسکوئید به شمار می رود که به خوبی خطاها و هشدارهای احتمالی ومحل وقوع آنها را در

خروجی گزارش می کند.

رفع اشکال کامل اسکوئید همراه با جزییات

فرمان squid -d که در بالا تشریح شد مراحل راه اندازی اسکوئید را همراه با نمایش خطاهای احتمالی نمایش می داد در صورتیکه بخواهیم اطلاعات کامل تری در مورد قسمت های محتلف فایل پیکربندی و اطمینان از اینکه کلیه ی تنظیمات صحیح و بدون اشکال هستند کسب کنیم باید از دستور squid -X استفاده کنیم.این دستور کلیه ی تنظمات موجود را به صورت خط به خط و به عبارتی موردی بررسی می کند و کوچترین خطا و یا انحرافی را نشان می دهد خروجی این دستور به صورت زیر است:

```
squid3 -X
2012/01/16 22:06:36.199| command-line -X overrides: ALL,7
2012/01/16 22:06:36.200| CacheManager::registerAction: registering legacy mem
2012/01/16 22:06:36.200| CacheManager::findAction: looking for action mem
2012/01/16 22:06:36.200| Action not found.
2012/01/16 22:06:36.200| CacheManager::registerAction: registered mem
2012/01/16 22:06:36.200| CacheManager::registerAction: registering legacy squidaio_counts
2012/01/16 22:06:36.200| CacheManager::findAction: looking for action squidaio_counts
2012/01/16 22:06:36.200| Action not found.
2012/01/16 22:06:36.200| CacheManager::registerAction: registered squidaio_counts
2012/01/16 22:06:36.200| CacheManager::registerAction: registering legacy diskd
2012/01/16 22:06:36.200| CacheManager::findAction: looking for action diskd
2012/01/16 22:06:36.200| Action not found.
2012/01/16 22:06:36.200| CacheManager::registerAction: registered diskd
2012/01/16 22:06:36.200| Detected IPv6 hybrid or v4-mapping stack...
```

```
2012/01/16 22:06:36.200| IPv6 transport Enabled
2012/01/16 22:06:36.200| aclDestroyACLs: invoked
2012/01/16 22:06:36.200| ACL::Prototype::Registered: invoked for type src
2012/01/16 22:06:36.200| ACL::Prototype::Registered:    yes
2012/01/16 22:06:36.200| ACL::FindByName 'all'
2012/01/16 22:06:36.200| ACL::FindByName found no match
2012/01/16 22:06:36.200| aclParseAclLine: Creating ACL 'all'
2012/01/16 22:06:36.200| ACL::Prototype::Factory: cloning an object for type 'src'
2012/01/16 22:06:36.200| aclIpParseIpData: all
2012/01/16 22:06:36.200| aclIpParseIpData: magic 'all' found.
2012/01/16 22:06:36.200| aclParseAclList: looking for ACL name 'all'
2012/01/16 22:06:36.200| ACL::FindByName 'all'
2012/01/16 22:06:36.200| Processing Configuration File: /etc/squid3/squid.conf (depth 0)
2012/01/16 22:06:36.202| Processing: 'http_port 3128 transparent'
```

خروجی این دستور کلیه ی acl های موجود ،کلاس های ip ، نام میزبان و سایر مراحل را به طور مرتب بررسی می کند.

تجزیه ی تنظیمات اسکوئید به منظور یافتن اشکالات احتمالی

از آنجایی که یافتن خطاهای موجود و اشکال زدایی آنها یکی از مهمترین عوامل کارکرد صحیح و بدون نقص اسکوئید به شمار می رود دستورات متنوعی بدین منظور در نظر گرفته شده است.دستوراتی که در بالا معرفی شدند همگی گزارشات نسبتا کاملی از کلیه ی تنظیمات موجود به دست می دهند در صورتکه بخواهیم دامنه این گزارشات را محدود تر کنیم می توانیم از دستوراتی که بیشتر جنبه ی تجزیه ای دارند استفاده کنیم .فرمان squid -k parse خطاهای احتمالی و هشدارهای امنیتی ناشی از به کارگیری نادرست یک دستور و یا احتمال نفوذ در اثر اجرای ناقص یک دستور را به ما گوشزد می کند و از طولانی کردن گزارش جلوگیری می کند.برای پی بردن به نحوه عملکرد آن ،خط false_directive 500 را به فایل پیکربندی squid.conf اضافه می کنیم و پس از ذخیره کردن آن دستور squid -k parse را اجرا می کنیم، به خروجی

دستور توجه کنید:

```
squid -k parse
2012/01/16 22:35:15| cache_cf.cc(364) parseOneConfigFile: squid.conf:1 unrecognized: 'false_directive'
```

خروجی این دستور به ما می گوید که این خط دستور نامتعبر است و باید اصلاح شود.بنابراین درصورتکیه ما در هر جای فایل پیکربندی مرکتب خطایی شویم با اجرای مرتب این دستور می توانیم محل دقیق بروز خطا را پیدا کرده و نسبت به اصلاح آن اقدام کنیم بدون اینکه نیازی به راه اندازی مجدد اسکوئید داشته باشیم.

ارسال سیگنالهای مختلف به سوی اسکوئید در حال اجرا

با استفاده از فرمان k- میتوان سیگنالهای کنترلی مختلفی را به سوی اسکوئید در حال فعالیت ارسال نمود.با استفاده از این دستور میتوان پیام های توقف،راه اندازی دوباره،بازخوانی دوباره فایل پیکربندی،و یا گرد سازی فایلهای ثبت رویداد و بسیاری دیگر از فرمانهای قابل اجرا را به سوی اسکوئید ارسال کرد.

بازخوانی مجدد فایل پیکربندی

زمانیکه تغییری در تنظیمات پیکربندی صورت می گیرد برای اینکه این تغییرات اعمال شوند فایل پیکربندی اسکوئید باید دوباره فراخوانی شود زیرا این فایل یک بار و آن هم زمانیکه اسکوئید راه اندازی می شود فراخوانی شده و تنظیمات جاری مطابق آن اعمال می شود.راه اندازی مجدد اسکوئید یا ریستارت کردن آن موجب ایجاد تاخیر ۳۰ ثانیه ای و بستن تمامی سوکتهای ارتباط کاربران به طور موقت می شود این کارممکن است موجب ناخشنودی کاربران شبکه شود برای جلوگیری از این کار می توان دستور squid -k reconfigure را به کار گرفت ، این دستور فایل پیکربندی را بدون راه اندازی مجدد اسکوئید فراخوانی کرده و تغییرات ایجاد شده را اعمال می نماید و براحتی می توان از ایجاد کوچکترین وقفه در شبکه جلوگیری به عمل آورد.همچنین استفاده از فرمان squid -k parse قبل از فرمان reconfigure موجب می شود که ما از صحیح بودن تغییرات اعمال شده اطمینان حاصل نماییم.

*نکته:در صورتکیه از بسته های از پیش کامپایل شده استفاده می کنید می توانیداز دستورات /**etc/init.d/squid reload** و یا **service squid reload** که معادل دستور **reconfigure** هستند استفاده کنید.*

متوقف کردن فعالیت اسکوئید

برای موقف کردن فعالیت اسکوئید در حال اجرا باید سیگنال توقف فعالیت یا shutdown را به سوی اسکوئید ارسال کرد برای این کار دستور squid -k shutdown را در ترمینال سرور وارد کنید .این دستور کلیه ی فعالیت های اسکوئید را متوقف کرده و تمامی سوکتهای ارتباطی برقرار شده را می بندد .

توقف سریع اسکوئید

در شبکه های بزرگ که تعداد کاربران موجود در شبکه زیاد هستند زمان مورد نیاز برای خاتمه ی تمامی ارتباطات و بستن تمامی سوکتهای ارتباطی زمان بر است در صورتکیه بخواهیم اسکوئید را سریعا متوقف کنیم ، از دستور squid -k interrupt استفاده می کنیم. در برخی موارد به دلیل برخی مشکلات نظیر کمبود منابع سخت افزاری دستورات k -shutdown یا k reconfigure- قادر به متوقف کردن اسکوئید نیستند و به رغم ارسال سیگنالهای توقف فعالیت ، اسکوئید همچنان به فعالیت خود ادامه می دهد در این گونه موارد فرمان k kill- می تواند این مشکل را برطرف کند:

```
squid -k kill
```

این دستور از طریق ارسال سیگنال kill به اسکوئید به اجبار باعث توقف کلیه ی فعالیت های آن می شود .

بررسی وضعیت فعالیت اسکوئید

به منظور حصول اطمینان از اینکه اسکوئید فعال است یا خیر فرمان k check- به کار می رود این دستور وضعیت جاری اسکوئید را از طریق proccess id چک می کند در صورتکیه اسکوئید فعال باشد و مشکلی در فعالیت مشاهده نشود این دستور چیزی بر نمی گرداند در غیر اینصورت دلیل بروز خطا را در صفحه نمایش نشان می دهد. به پیام خطای زیر توجه کنید:

```
squid: ERROR: Could not send signal 0 to process 25243: (3) No such process
```

این خطا بیان می کند که اسکوئید فعال نیست و دستور مورد نظر قادر به ارسال چنین سیگنالی به سوی اسکوئید نمی باشد.

گرد سازی فایلهای ثبت رویداد

طی یک دوره ی زمانی مشخص با توجه به میزان ترافیک شبکه و تعداد کاربران فایلهای ثبت رویداد که کوچکترین فعالیت موجود در شبکه را ثبت می کنند حجیم می شوند و فضای قابل توجهی از دیسک سخت را اشغال می کنند .به منظور جلوگیری از پر شدن بی مورد فضای ارزشمند ذخیره سازی دستور rotate ایجاد شده است .هنگامی که اسکوئید این

سیگنال را دریافت می کند فرایندثبت رویداد های شبکه را متوقف می کند و فایلهای ثبت رویداد را بسته به نوع تنظیمات اعمالی صورت گرفته در squid.conf حذف و یا تغییر نام می دهد و آنها را تا حد امکان گرد می کند . تنظیمات اعمالی بر این فرمان از طریق برچسب logfile_rotate در squid.conf صورت می گیرد.پس از انجام عمل گرد سازی، فایلهای ثبت رویداد دوباره برای نوشتن باز می شوند.گرد شازی این فایلها به صورت دستی کاری تکراری و خسته کننده است ، برای حل این موضوع بهتر است فرایند انجام این دستور را در صف فعالیتهای سیستم عامل یا cron job قرار دهیم تا این عمل به صورت خودکار و بدون دخالت ما انجام شود به مثال زیر توجه کنید:

```
59 23 * * * /opt/squid/sbin/squid -k rotate
```

این دستور فرایند گرد سازی فایلها را در پایان نیمه ی هر شب انجام می دهد.در مورد فایلهای ثبت رویداد ، نحوه عملکرد آنها و نحوه مدیریت آنها در فصلهای آینده به طور مفصل بحث خواهیم کرد..

بازسازی مخزن ذخیره سازی داده ها

پس از هر بار راه اندازی ،اسکوئید مخزن ذخیره سازی داده ها را فراخوانی می کند در صورتکیه به هر دلیلی اسکوئید نتواند این فرایند را به درستی انجام دهد و همزمان کاربران نیز درخواستهایی را به سوی آن ارسال کنند ، اسکوئید فرایند فراخوانی مخزن ذخیره سازی و پاسخ به درخواست کاربران را به طور همزمان انجام می دهد که نتیجه ی این کار کند شدن عملکرد اسکوئید است.برای جلوگیری از این مشکل دستور squid -F اسکوئید را مجاب می کند تا زمانیکه فرایند فراخوانی داده های ذخیره شده را به تمام نرساند درخواستی را از سوی کاربران نپذیرد این عمل موجب می شود فراخوانی اطلاعات با سرعت و به طور صحیح انجام شود.این کار در کش سرورهایی که فضای ذخیره سازی زیادی دارند و ترافیک سنگینی دارند بسیار مهم است.

بررسی دوباره مخزن ذخیره سازی داده ها

دستور –F اسکوئید را مجاب به بررسی مخزن نگهداری داده های کاربران به منظور رفع مشکلات احتمالی می کند.با استفاده از دستور –S میتوان این فرایند را به صورت دوبار در آن واحد تبدیل کرد،به طوریکه اسکوئید مخزن داده ها را به دقت بسیار بیشتر بررسی می کند.نحوه استفاده از این دستور به صورت زیر است:

```
squid -S -d 1
```

با اجرای دستور بالا خروجی دستور به صورت زیر خواهد بود:

```
2010/07/21 21:29:22| Beginning Validation Procedure
```

```
2010/07/21 21:29:22| UFSSwapDir::doubleCheck: SIZE MISMATCH
2010/07/21 21:29:22| UFSSwapDir::doubleCheck: ENTRY SIZE: 1332
SIZE: 114
2010/07/21 21:29:22| UFSSwapDir::dumpEntry: FILENO 00000092
2010/07/21 21:29:22| UFSSwapDir::dumpEntry: PATH /squid_
cache/00/00/00000092
2010/07/21 21:29:22| StoreEntry->key: 0060E9E547F3A1AAEEDE369C
2010/07/21 21:29:22| StoreEntry->next: 0
2010/07/21 21:29:22| StoreEntry->mem_obj: 0
2010/07/21 21:29:22| StoreEntry->timestamp: 1248375049
2010/07/21 21:29:22| StoreEntry->lastref: 1248375754
2010/07/21 21:29:22| StoreEntry->expires: 1279911049
2010/07/21 21:29:22| StoreEntry->lastmod: 1205097338
2010/07/21 21:29:22| StoreEntry->swap_file_sz: 1332
2010/07/21 21:29:22| StoreEntry->refcount: 1
2010/07/21 21:29:22| StoreEntry->flags: CACHABLE,DISPATCHED
2010/07/21 21:29:22| StoreEntry->swap_dirn: 0
2010/07/21 21:29:22| StoreEntry->swap_filen: 146
2010/07/21 21:29:22| StoreEntry->lock_count: 0
2010/07/21 21:29:22| StoreEntry->mem_status: 0
...
```

در این حالت اسکوئید تمام تلاش خود را برای بررسی دقیق اطلاعات موجود در مخزن کش انجام می دهد و تمامی اطلاعات ذخیره شده را از حیث سطح اعتبار بررسی می کند.

راه اندازی خودکار اسکوئید

برای راه اندازی خودکار اسکوئید در مواردی که سیستم عامل دوباره راه اندازی می شود دو راه وجود دارد یک راه این است که به صورت دستی فرمان /usr/local/squid/sbin/squid و یا service squid start را اجرا کنیم اما راه حل بهتر اجرای خودکار اسکوئید است .در بسیاری از سیستم عاملهای مشهور لینوکس و یونیکس فایل اسکریپتی به نام /etc/rc.local وجود دارد که از آن برای نگهداری دستورات init استفاده می شود بدین منظور دستور /usr/local/squid/sbin/squid و service squid start را در این فایل قبل از کد exit 0 قرار می دهیم با این کار سیستم عامل پس از هر بار راه اندازی این فایل را بررسی می کند و دستورات موجود در آن را اجرا می کند.

خلاصه فصل

در این فصل دستورات مختلفی که برای راه اندازی و رفع اشکال اسکوئید به کار می روند را بررسی کردیم و نحوه به کارگیری آنها و همچنین تفاوت و ویژگیهای هر یک و همچنین نحوه تغییر نام فایل پیکربندی اصلی و بررسی و رفع اشکال سریع و خطا به خط آن را آموختیم.در این فصل نحوه گرد سازی فایلهای ثبت رویداد و فراخوانی مجدد فایل پیکربندی بدون نیاز به راه اندازی مجدد شرح داده شد.به طور کلی مطالب مطرح شده در این فصل شامل موارد زیر است:

- تحلیل و بررسی فایلها پیکربندی اسکوئید به منظور یافتن اشکالات احتمالی
- استفاده از مناسبترین دستور به منظور رفع اشکال احتمالی فایل پیکربندی
- بازخوانی تنظیمات جدید پیکربندی توسط اسکوئید لدون نیاز به توقف فعالیت آن
- انجام خودکار فرایند گرد سازی فایلهای ثبت رویداد

دستورات مختلفی برای راه اندازی و کنترل اسکوئید و متوقف کردن آن در مواقع اضطراری تعیین شده اند که بسیاری از آنها در این فصل همراه با چگونگی به کارگیری آنها معرفی شدند.

حال که چگونگی راه اندازی اسکوئید را فراگرفتیم ،میتوانیم از Acl ها و سایر دستورات موجود در تنظیمات پیکربندی جهت کنترل و مدیریت شبکه ی خود استفاده کنیم.

فصل چهارم

به کارگیری ACL ها و Access Rule ها

در پایان مباحث فصل گذشته توانستیم یک پراکسی سرور مبتنی بر سرویس دهنده اسکوئید را نصب و راه اندازی کنیم در طول مباحث فصل جاری به مباحثی از قبیل تعریف Access list ها و تلفیق آن با http_access ها به منظور کنترل سطوح دسترسی به شبکه ، هدایت ترافیک شبکه به سوی گروهی خاص از کاربران ، جلوگیری از نفوذ و سوءاستفاده از شبکه توسط افراد ناشناس را بررسی خواهیم کرد .

به طور کلی اهداف این فصل به شرح زیر است:

- **بررسی انواع ACL**
- **بررسی انواع Access Rule ها**
- **ترکیب ACL و Access list ها**

آشنایی با Access Control List

Access Control Lists ها یکی از مهمترین عناصر موجود در اسکوئید به شمار می روند که به دلیل انعطاف پذیری بسیار بالای آنها امکان اعمال کنترل بر شبکه را برای ما فراهم می آورند .در فصل دوم در مورد نحوه ساخت ACL ها مطالب مفصلی ارائه شد اما اکنون زمان آن فرا رسیده است که بتوانیم این دستورات را در شبکه ی خود به کار ببندیم. به طور کلی ACL ها می توانند تمامی پردازش موجود در شبکه را کنترل کنند که این سیاستهای کنترلی خود را توسط Access Rule ها پیاده سازی می کنند .قبل از شروع کار با ACL ها کاربر می بایست نحوه کار با آدرس های ip و نحوه گروه بندی آدرس ها را از طریق Subnet ها آشنایی داشته باشد بدین منظور مراجع فراوانی وجود دارند که مباحث TCP/IP را به طور رایگان آموزش می دهند به طور مثال برای جهت کسب اطلاعات بیشتر در مورد مباحث تخصیص ip و Subneting به دانشنامه ی آزاد ویکی پدیا به نشانی http://en.wikipedia.org/wiki/Subnetwork#IPv4_subnetting مراجعه نمایید. برای شروع کار و نحوه ساخت یک ACL به مثال زیر توجه کنید:

```
acl clients src 192.0.2.0/25
```

در مثال بالا ساختار دستوری به چهار قسمت کلی تقسیم شده است :برچسب acl در ابتدای تعریف هر acl ی اجباری است و تغییر نمی کند بخش دوم مربوط به نامگذاری acl است که در اینجا clients انتخاب شده است انتخاب نام در این قسمت کاملا اختیاری است اما انتخاب واژه های مناسب با کاربرد هر یک در درک آسانتر آنها به ویژه در مواردی که چندین Acl داریم بسیار مفید خواهند بود بخش سوم دستور در واقع نوع acl را تعیین می کند که در اینجا src تعیین شده است ، تنظیمات این بخش در ادامه بررسی می شود و بالاخره بخش آخر که در واقع اطلاعات وارد شده توسط ما است بسته به نوع ACL تعریفی متفاوت خواهد بود مثلا در اینجا چون هدف ما تعریف محدوده ی ip بوده طبیعتا اطلاعات وارد شده هم می بایست از نوع ip باشد ، رعایت نکردن این موضوع موجب ایجاد خطا می شود. با توجه به تعاریف بالا در این مثال یک Aclبا محدوده ی آدرس 192.0.2.1 تا 192.0.2.127 تعریف شده است به طوریکه عبارت 192.0.2.0/25 نشان دهنده یک subnet با 127 آدرس ip می باشد.در قسمت بعدی این Acl عبارت src قرار گرفته است که یکی از انواع موجود Acl محسوب می شود و به منظور شناسایی آدرس ip منبع یک درخواست به کار می رود.

مقایسه ی ACL ها از نظر سرعت عملکرد

تمامی ACL های موجود در دو دسته ی عمده ی سریع و کند تقیسم بندی می شوند نوع سریع آنها برروی مقوله ی آدرس های وب ، آدرس های IP ،هدر های HTTP و سایر مواردی است که به طور کلی پردازش کمی را طلب می کنند و خروجی از آنها به سادگی صورت می گیرد و اسکوئید برای بررسی و تمایز داده ها به قدرت پردازشی کمتری نیاز دارد فعالیت می کنند .نوع کند این دستورات شامل داده هایی است که بار پردازشی به نسبت بیشتری نسبت به نوع قبل بر آنها تحمیل می شود، DNS

resolver ها و dst و srcdomain ها از جمله مواردی هستند که در دسته یACL های کند طبقه بندی می شوند.به کارگیری صحیح و به موقع این ACL ها در کارایی و پایداری و همچنین سرعت پرازش و تبادل داده نقش مهمی ایفا می کنند همیشه سعی کنید در صورتکیه لزومی برای استفاده از انواع کندتر ندارید از به کارگیری بی مورد آنها خودداری کنید.جهت آگاهی از فهرست تمامی Acl های سریع و کند به آدرس http://wiki.squid-cache.org/SquidFaq/SquidAcl#Fast_and_Slow_ACLs مراجعه فرمائید .

آدرس های ip مبدا و مقصد

هر درخواستی که از سوی یک کاربر به سوی اسکوئید ارسال می شود شامل اطلاعات مهمی مانند آدرس ip فرستنده و گیرنده،نشانی url ،آدرس سخت افزاری MAC و اطلاعات دیگری که توسط آنها فرستنده و گیرنده ی یک درخواست را مشخص می کند دونوع src , dst به منظور مشخص کردن آدرس گیرنده و ارسال کننده ی یک درخواست به کار می روند:

```
acl client src 192.0.2.25
acl clientd src 192.0.2.25/32
```

در این مثال در خط اول بعد از وارد کردن آدرس ip مقداری برای subnet تعیین ولی در خط دوم با همان آدرس مقدار۳۲ در نظر گرفته شده اما در واقع هر دو acl در یک محدوه ی آدرس قرار دارند و به عبارتی معادل یکدیگرند. . دلیل این موضوع این است که اسکوئید در مواقعی که محدوده ی آدرس مشخص نشده باشد به طور پیش فرض مقدار ۳۲ یعنی بیشترین تعداد ip لحاظ می شود.در ساده ترین حالت نیازی به تعیین subnet نیست اما در صورتیکه بخواهیم تعداد آدرس های مورد استفاده رامحدود کنیم راهی جزء استفاده از Subnet ها نیست :

```
acl  lab  src 192.0. 3.0/27
```

در acl بالا تعدادآدرس های مورد استفاده را در محدوده 192.0.3.0 تا 192.0.3.31 مشخص می کند.نحوه ساخت acl های نوع dst نیز کاملا شبیه موارد بالا هستند به مثال زیر توجه کنید:

```
acl website dst 198.51.100.86
```

در این مثال تمامی درخواستهای به مقصد 198.51.100.86 شناسایی و ردیابی می شوند.

نکته: Acl های src و dstبه ترتیب از نوع Acl های سریع و کند به شمار می روند.

ساخت acl های پیچیده تر

تاکنون تمامی مثالهایی که طرح شد همگی از یک محدوده ی آدرس تشکیل شده بودند اما فرض کنید در یک شرکت نسبتا بزرگ چندین محدوده ی ip برای واحدهای مختلف تعیین شده است و هر بخش به صورت مجزا دارای آدرسهای منحصر به فرد هستند بنابر قاعده ی بالا برای هر محدوده ی آدرس می بایست یک Acl مجزا تعریف شود:

10.1.2.0/24 #org1

10.1.3.0/24 #org2

10.1.4.0/24 #org3

10.1.5.0/24 #org4

10.1.6.0/24 #org5

برای اینکه بتوانیم تمامی آدرسهای بالا را در یکAcl قرار دهیم از روش زیر کمک می گیریم:

acl mkt_dept src 10.1.2.0/24 10.1.3.0/24 10.1.4.0/24 10.1.5.0/24 10.1.6.0/24

دراین روش به راحتی تمامی آدرسها را در یک Acl تعریف می کند اما راه ساده تر به شکل زیر است:

acl mkt_dept src 10.1.2.0-10.1.6.0/24

این دستور کوتاه معادل مثال بالا است در این روش طول Acl بسیار کوتاه تر و نوستن آن آسانتر است به هر حال این دو روش کار یکسانی را انجام می دهند و آن هم تعریف چند محدوده ی ip در یک Acl و در کنار هم ، هر دو روش بالا صحیح هستند و شما بسته به اینکه کدام روش برای شما آسان تر است آن را انتخاب کنید.همانطور که می بینید دستورات Acl در اسکوئید علی رغم ساختار بسیار ساده ی خود از کارایی و هوشمندی بسیار بالایی برخوردار هستند به طوریکه در مواقعی که شما مایل به محدود کردن محدوده ی آدرس نیستید Aclتعریف شده به صورت خودکار طور محدوده مناسب را تشخیص داده و subnet مناسب با آن را در نظر می گیرد.

شناسایی نشانی IP های محلی

یکی دیگر از انواع Acl هایی که در این بخش معرفی می شوند myip نام دارد .به کارگیری این نوع فقط زمانی مفید است که سروری که اسکوئید برروی آن درحال اجراست ، بیش از یک کارت شبکه داشته باشد.به طور مثال در صورتکیه پراکسی سرور ما آدرس های 192.0.2.25, 198.51.100.25 و یک آدرس IP عمومی دیگر را در اختیار داشته باشد حال فرض کنیم کاربران بخش تحقیقات سازمان از آدرس 198.51.100.25 به مظنور ارتباط با پراکسی سرور استفاده می کنند و پرسنل بخش آزمایشگاه نیز از آدرس 192.0.2.25 جهت برقراری ارتباط با پراکسی سرور اسکوئید استفاده کنند در این صورت میتوانیم

Acl ها مورد نیاز خود را به صورت زیر تعریف کنیم:

```
acl research_center_net myip 198.51.100.25

acl student_lab_ip myip 192.0.2.25
```

حال می میتوانیم مانند دو پراکسی سرور مجزا خدمات متفاوتی را به دو محدوده ی متفاوت IP برروی دو کار شبکه ی مختلف در اختیار کاربران قرار دهیم.به هر حال در بسیاری از مواقع از نشانی های IP به منظور شناسایی کاربران استفده می کنیم اما موارد نادری نیز وجود دارند که در آن به جای IP از آدرس های سخت افزاری MAC جهت شناسایی و مدیریت کاربران استفاده می شود که در ادامه به معرفی آن می پردازیم.

استفاده از آدرس های MAC در ACL ها

آدرس های سخت افزاری(MAC (Media Access Control مانند آدرسهای ip برای هر ماشین منحصر به فرد هستند از آدرسهای سخت افزاری موجود در شبکه نیز می توان در ACL ها استفاده کرد و کاربران را توسط این آدرسها شناسایی و کنترل کرد این آدرسها مستقیما به کارت شبکه ی ماشین مورد نظر تخصیص داده می شود. Acl هایی که برای استفاده همراه با آدرسهای MAC در نظر گرفته شده از نوع arp هستند و نحوه ساخت و به کارگیری آنها دقیقا مانند موارد قبل است به مثال زیر توجه کنید:

```
acl mac_acl arp 00:1D:7D:D4:F3:EE
```

در اینجا مک آدرس 00:1D:7D:D4:F3:EE موجود در شبکه شناسایی می شود توجه داشته باشید شرط استفاده از این نوع acl ها فعال کردن ویژگیهای enable-eui – یا --enable-arp-acl هنگام کامپایل اسکوئید است .توجه داشته باشید این نوع از Acl در تمامی سیستم عاملها پشتیبانی نمی شوند و همواره قبل از به کارگیری آن از فراهم بودن مقدمات آن در سیستم عامل مورد استفاده اطمینان حاصل نمائید.

نامهای دومین مبدا و مقصد

شناسایی مبدا درخواستها از طریق نشانی های IP کاربران آسان است زیرا این آدرس ها از طریق خود ما در Acl ها تعریف شده و در اختیار کاربران قسمتهای مختلف شبکه قرار می گیرند اما در مقابل شناسایی درخواستها از طریق آدرسهای مقصد به راحتی استفاده از IP ها نیست زیرا:

- نشانی IP یک میزبان خارجی ممکن است تغییر کند

- ترجمه ی آدرس های مقصد (آدرس های وب) به IP مورد نظر آنها به دلیل استفاده از DNS سرور ها باعث ایجاد تاخیر جزئی در شبکه می شود.

اسکوئید دو Acl به نام های srcdomain و dstdomain را به ترتیب به منظور ساخت Acl های بر پایه ی مبدا و مقصد آدرسهای دامنه فراهم آورده است که به ترتیب جزء Acl های کند و سریع محسوب می شوند.در ادامه یک Acl به منظور شناسایی درخواستهای به نشانی دامنه www.example.com می سازیم:

acl example dstdomain www.example.com

این Acl تمامی درخواستهایی را که جزء دامنه example.com باشند را شناسایی می کند بنابراین در صورتیکه ما در مرورگر خود آدرسهای http://www.example.com و یا http://www.example.com/index.html را وارد نمائیم ، این Acl تمامی این درخواستهای را شناسایی می کند زیرا هر دوی آنها جزء دامنه example.com محسوب می شوند.یکی از مشکلات این روش این است که در این نوع Acl آدرسهایی نظیر http://example.com و یا http://some.example.com شناسایی نمی شوند بنابراین در صورتکیه در مرورگر خود آدرس http://video.example.com را که یک زیر دامنه example.com محسوب می شود را وارد کنیم ، این درخواست شناسایی نخواهد شد و از محدوده ی Acl تعریف شده خارج می شود.برای غلبه بر این مشکل کافی است در Acl تعریف شده قبل از تعریف نشانی دامنه مورد نظر یک نقطه (.) اضافه کنیم در این صورت اسکوئید و Acl تعریف شده تمامی زیردامنه های مربوط به نشانی تعریف شده را شناسایی می کنند.به مثال زیر توجه کنید:

acl example dstdomain .example.com

در اینجا Acl تعریف شده تمامی زیردامنه های مربوط به نشانی example.com مانند video.example.com، news.example.com و یا www.example.com را شناسایی می کند.به طور مشابه در صورتیکه یک Acl مانند acl example_uk dstdomain .uk.example.com تعریف کنیم در این صورت تمامی زیردامنه های مربوط به uk.example.com شناسایی می شوند اما توجه داشته باشید این دستور دامنه های example.com را تشخیص نخواهد داد.به مثال دیگری در این زمینه به صورت زیر است:

acl our_network srcdomain .company.example.com

این Acl تمامی درخواستهایی که به نوعی جزء زیردامنه company.example.com باشند را شناسایی و تشخیص می دهد.در این بخش چگونگی استفاده از Acl های نوع srcdomain را آموختیم.نکته ی مهمی که در اینجا مطرح است این است که این نوع Acl جزء Acl های کند به شمار می رود و در استفاده از آن تمامی جوانب ممکن مانند استفاده از DNS سرور های با کیفیت مناسب را مد نظر قرار دهید.

مدیریت هوشمندانه ی آدرسهای دامنه

دستورات کنترلی بالا برای شناسایی محدود چند آدرس به خوبی عمل می کنند اما شناسایی و کنترل چند صدو یا هزاران آدرس کاری بسیار دشوار و بخسته کننده است از آنجا که اسکوئید همواره برای هر مشکلی راه حل مناسبی ارائه می دهد برای رفع این مشکل نیز دو acl به نامهای srcdom_regex و dstdom_regex را فراهم کرده است. این دو برخلاف موارد قبلی که عملکرد صحیح آنها وابسته با آدرس کامل دامنه مورد نظر بود این دستورات حساس به کلمات هستند و عباراتی که در قسمت تعاریف Acl می گیرد در تمامی آدرسها جستجو می شود و در صورتیکه یک آدرس شامل آن عبارت باشد مطابق سیاست های آن Acl با آن رفتار می شود. به مثال زیر توجه کنید:

```
acl bad_sites  dstdom_regex -i  evil

http_access deny bad_sites
```

در اینجا هر آدرسی که عبارت evil در نشانی دامنه آن وجود داشته باشد مسدود می شود. این روش یکی از بهترین روش های کنترل بر محتوای وب می باشد که می توان بر کاربران یک مجموعه ی خاص اعمال کرد در واقع در این روش فارغ از اینکه آدرس مورد نظر مربوط به کدام کشور است و یا آدرس اصلی و یا زیر مجموعه ی آدرس دیگری است اعمال کنترلی شما را پیاده سازی می کند .نوع srcdom_regex نیز مشابه مورد بالا رفتار می کند.بدین ترتیب با قرار دادن کلماتی که مغایر سیاست های کنترلی شما محسوب می شود از دسترسی کاربران به آنها جلوگری به عمل می آید.

درگاه مقصد

هرگاه کاربری درخواستی را به طرف اسکوئید ارسال می کند این درخواست باید ازطریق کانالهای ارتباطی مشخصی که در مبدا و مقصد تعریف شده و قابل شناسایی هستند عبور کند ، به این درگاهها Port گفته می شود .درگاههای جهانی ارتباطی به صورت استاندارد تعریف شده اند و در تمامی گستره ی شبکه ی جهانی اینترنت یکسان است مثلا درگاه پروتکل http شماره ی ۸۰ است بدین معنی که تمامی درخواستهای http ای که به سوی اسکوئید ارسال می شود از طریق درگاه http به مقصد می رسد برای اینکه یک ارتباط موفقیت آمیز باشد می بایست هر دو طرف مبادله ی کننده ی اطلاعات، داده های خود را از طریق یک کانال ارتباطی یکسان ارسال کنند.برای مثال در صورتیکه کاربری نشانی www.example.com را در مرورگر خود وارد کند ،برای اینکه اسکوئید قادر به دریافت این درخواست باشد ،باید هم کاربر و هم اسکوئید از یک شماره ی د

درگاه یکسان جهت تبادل اطلاعات استفاده کنند..port نام Acl ی است که اسکوئید از آن برای تعریف انواع درگاهها و کنترل بر آنها استفاده می کند:

```
acl allowed_port port 80
```

در اینجا درگاه شماره ی ۸۰ به عنوان یکی از درگاههای مجاز تعیین شده است توجه به این نکته ضروری است که تمامی درگاههای تعریف شده می بایست عددی بین ۰ تا ۶۵۵۳۵ باشد.

Acl های تعریف شده از نوع port میتوانند چندین درگاه را شامل شود یا محدوده ی از درگاه ها را تعیین کرده و اجازه و یا عدم اجازه ی عبور داده ها را برای آنها تعیین کرد:

```
acl allowed_ports port 80 443 1025-65535
```

در این Acl درگاههای ۸۰, ۴۴۳ و تمامی درگاههای موجود میان ۱۰۲۵ تا ۶۵۵۳۵ تعریف شده است در واقع هدف از این کار شناساندن درگاههای مورد نظر به اسکوئید و همچنین شناسایی و ارسال درخواستهای کاربران به آن می باشد.

برای جلوگیری از حملات و سوءاستفاده های احتمالی همواره توصیه می شود که فقط به درگاههای مورد نیاز اجازه ی برقراری ارتباط داده شود و مابقی آنها مسدود شوند زیرا باز گذاشتن یک درگاه بدون دلیل موجه می تواند امنیت شبکه را با خطر مواجه سازد. درفایل پیکربندی اسکوئید به طورپیش فرض درگاههای از پیش تعریف شده ی بسیاری وجود دارد که در بیشتر موارد موارد نیازی به اضافه کردن درگاههای درگیری نیست، safe_ports و ssl_ports دو مورد از Acl های از پیش تعریف شده است.خطوط زیر در فایل پیکربندی به صورت پیش فرض موجود است:

```
acl SSL_ports port 443     #https
acl Safe_ports port 80    #http
acl Safe_ports port 21    #ftp
acl Safe_ports port 443    #https
acl Safe_ports port 70     #gopher
acl Safe_ports port 210   #wais
acl Safe_ports port 280    # http-mgmt
acl Safe_ports port  488   # gss-http
acl Safe_ports port  591    # filemaker
acl Safe_ports port  777     # multiling http
acl Safe_ports port  1025-65535    # unregistered ports
```

در این مثال موارد آشنایی چون http , https , ftp مشاهده می شوند به این ترتیب شما نیز می توانید شماره ی درگاههای

مورد نظر خود را به آن اضافه نمائید اما همواره قبل از انجام این کار از مطمئن بودن درگاه مربوطه اطمینان حاصل نمایید و بالطبع مورادی که به آنها نیازی ندارید را غیر فعال کنید.در واقع این قسمت یکی از مهمترین موارد امنیتی موجود در اسکوئید به شمار می رود که تاثیر بسزایی در امنیت شبکه دارد.اکنون برای غیر فعال کردن و یا به عبارتی مسدود کردن سایر درگاههایی که در فهرست بالا نیستند به صورت زیر عمل می کنیم:

```
http_access deny !Safe_ports
```

به علامت ! قبل از safe_ports دقت کنید، این علامت تمامی درگاههایی که در لیست Acl پیش فرض safe_ports وجود نداشته باشد را مسدود خواهد کرد و به راحتی و بدون نیاز به تعریف دوباره ی Acl های مجزا می توان درگاههای ارتباطی باقی مانده را مسدود نمود.بدین ترتیب سرورهای خارجی مقصد فقط از طریق این درگاهها قادر به برقراری ارتباط با اسکوئید خواهند بود و تبادل اطلاعات از طریق سایر درگاههای غیرمجاز میسر نخواهد بود.

نام درگاه محلی

اسکوئید علاوه بر port ، نوع دیگری از Acl به نام myportname را فراهم آورده است این نوع Acl برای استفاده ی کاربرانی است که در شبکه ای که اسکوئید در آن قرار دارد فراهم شده است Myportmane مانند myip تنها زمانی مفید است که در قسمت تعاریف http_port بیش از یک درگاه تعریف شده باشد در واقع بدین وسیله می توان برای اسکوئید بیش از یک درگاه و ip تعریف کرده و در اختیار گروه مختلفی از کاربران به صورت کاملا مجزا قرار داد:

```
http_port 192.0.2.21:3128 name=research_port
http_port 192.0.2.25:8080 name=student_port
acl research_lab_net myportname research_port
acl student_lab_net myportname student_port
```

توسط این تعاریف می توان گروههای مختلف کاربران با Subnet های گوناگون را به صورت کاملا موثر کنترل نمود و آدرسهای با کلاسهای متفاوت را همراه با درگاه اختصاصی برای کاربران در کلاسهای مختلف ip تعریف نمود در اینجا دو آدرس با درگاه متفاوت تعریف شده است و به صورت دو شبکه ی مجزا عمل می کنند.

متدهای HTTP

هر درخواست http که به سوی سرورهای مقصد ارسال می شود شامل متد های متخلف HTTP هستند برای مثال هنگامی که ما

در مرورگر خود آدرس example.com را وارد می کنیم متد GET را به سوی وب سرور مورد نظر ارسال می کنیم همچنین هنگامی که یک فرم آنلاین را تکمیل می کنیم متد POST را برای ارسال اطلاعات به سوی سرور مورد نظر ارسال می کنیم.به طور کلیDELETE , GET و CONECT از جمله متد های مورد استفاده در پروتکل HTTP هستند .اسکوئید از Acl نوع method به منظور شناسایی و تعریف متد های مختلف بهره می گیرد. به طور پیش فرض تمامی متد های موجود به جز متد CONECT اجازه برقراری ارتباط و تبادل داده را دارند دلیل محدود شدن نوع CONECT استفاده از آن برای ایجاد Tunnel و احیانا خروج از کنترل پراکسی سرور است در تنظیمات پیش فرض پیکربندی نیز به این نکته توجه شده و سطح دسترسی به متد Connect محدود شده است.

```
acl CONNECT method CONNECT

acl SSL_ports port 443

http_access deny CONNECT !SSL_ports
```

توجه داشته باشید که Acl های نوع Connect با متد Conncet کاملا متفاوت هستند و وجه اشتراک آنها فقط تشابه اسمی آنهاست اما عملکرد آنها به طور کلی متفاوت است.به طور پیش فرض اسکوئید فقط به ارتباطات رمزنگای شده ی HTTP SSLبرروی درگاه 443 که در واقع نوع استاندارد ارتباطات امن HTTPS است اجازه ی برقراری ارتباط با متد Connect را فراهم می کند و تمامی ارتباطات دیگر از سایر درگاهها را مسدود می نماید.

نکته: همواره با اضافه کردن درگاههای مورد نظر خود به **SSL_PORTS** *می بایست این درگاهها به* **Acl** *نوع* **SSL_ports** *نیز اضافه کنید تا امکان برقراری ارتباط میسر شود.*

کنترل و شناسایی درخواستها با استفاده از پروتکل ها درخواست شده

درخواستهای دریافت شده توسط اسکوید از طریق نوع پروتکل به کاررفته در ارتباط شناسایی می شود به طور مثال هنگامی که آدرس http://example.com به سوی اسکوئید ارسال می شود ،اسکوئید از طریق هدر http و قالب هدر ارسال شده نوع ارتباط را از نوع http شناسایی می کند همچنین در مورد سایر پروتکل های ارتباطی نظیر ftp مثلا آدرس ftp://example.com ،اسکوئید باز هم از طریق قابل هدر و همچنین عبارت ftp در ابتدای آدرس نوع ارتباط برقرار شده را FTP تشخیص می دهد. proto نام acl ی است که از آن برای شناسایی و کنترل درخواستها از طریق پروتکل ها به کار می رود سایر قالب های URL شامل https , urn, gopher و whoisمی باشد.

فرض کنیم می خواهیم درخواستهای FTP ارسال شده از سوی یک subnet خاص مسدود شود برای این کار تنظیمات زیر را به فایل پیکربندی اضافه می کنیم:

```
acl ftp_requests proto FTP
acl research_labs src 192.0.2.0/24
http_access deny research_labs ftp_requests
```

در اینجا ارتباط کلیه ی کاربران موجود در محدوده ی آدرس ۱۹۲.۰.۲.۰/۲۴ جهت تبادل اطلاعات از طریق FTP مسدود می شود .این روش یکی از بهترین و موثرترین روش های کنترل بر سطح دسترسی کاربران از طریق پروتکل های ارتباطی است.

نکته: برخی از دیوارهای آتش موجود به طور پیش فرض ارتباطات **FTP** *فعال را مسدود می کنند برای کسب اطلاعات بیشتر در این باره به نشانی* **http://www.ncftp.com/ncftpd/doc/misc/ftp_and_firewalls.html** *مراجعه نمائید.*

تنظیمات دسترسی به Cache_mgr

cache_mgr.cgi یکی از ابزارهای کنترل و مانیتورینگ اسکوئید از طریق محیط گرافیکی است که همواره با بسته های مختلف اسکوئید عرضه می شود به دلیل امکان اعمال تنظیمات در برخی از قسمت های فایل پیکربندی اصلی اسکوئید به طور پیش فرض دسترسی به این رابط گرافیکی را برای تمامی اعضای شبکه مسدود می کند که این کار از طریق دستورات زیر انجام می شود:

```
acl manager proto cache_object
acl localhost src 127.0.0.1/32
http_access allow manager localhost
http_access deny manager
```

با توجه به این دستوارت فقط خود ماشین اجرا کننده ی اسکوئید قادر به نمایش cache_mgr است . برای اینکه بتوانیم به این رابط گرافیکی بروری سایر کامپیوترهای موجود در شبکه و یا حتی خارج از آن دسترسی داشته باشیم می بایست تنظیمات بالا را به صورت زیر تغییر دهیم:

```
acl manager proto cache_object
acl localhost src 127.0.0.1/32
acl admin_machines src 192.0.2.86 192.0.2.10
http_access allow manager localhost
```

http_access allow manager admin_machines

http_access deny manager

در اینجا علاوه بر خود سرور ، آدرسهای ۱۹۲.۰.۲.۱۰۲.۸۶ و ۱۹۲.۰.۲.۱۰ نیز امکان دسترسی به cache_mgr را خواهند داشت.

ACL های زمانی

ACL های زمانی یکی از جالبترین انواع ACL به شمار می روند،در این نوع ACL های تعریف شده در یک زمان خاص که توسط ما تعیین می شود شروع به کار کرده و در یک زمان مشخص نیز متوقف می شوند. در واقع در این نوع ACL مورد نظر ما در یک زمان های مشخصی از روز به فعالیت می پردازد و پس از آن متوقف می شود.Acl های زمانی باعث انعطاف پذیری بسیار زیاد ACL ها می شوند.نحوه ساخت آن از قاعده ی زیر پیروی می کند:

acl ACL_NAME time [day-abbreviation] [h1:m1-h2:m2]

تعیین مقدار برای روزهای هفته و بازه ی زمانی اختیاری است اما حتما یکی از آنها باید مشخص شوند.برای آسانتر شدن کار روزهای هفته را توسط یک حرف به اختصار به کار می رود. درجدول زیر نام روزهای هفته و نام اختصاری آنها را مشاهده می کنید:

روزهای هفته	نام مستعار
Sunday	S
Monday	M
Tuesday	T
Wednesday	W
Thursday	H
Friday	F
Saturday	A
All Weekdays	D

بدین ترتیب برای اجرای یک acl خاص در روزهای مشخصی از هفته بصورت زیر عمل می کنیم:

```
acl days time SMW

acl morning_hrs time MTWHF 09:00-12:59

acl lunch_hrs time D 13:00-13:59

acl evening_hrs time MTWHF 14:00-18:00
```

روزهای یکشنبه ،دوشنبه و چهارشنبه روزهایی هستند که acl مورد نظر فعال می شود.کلماتی که به عنوان مخفف روزهای هفته انتخاب شده اند می بایست درکنار هم و بدون فاصله قرار گیرند همچنین توجه داشته باشید که همواره زمان تعیین شده در بخش اول باید کمتر از بخش دوم باشد یعنی ۱۳:۵۰-۱۴:۳۰ نادرست است و درست آن بصورت ۱۴:۳۰-۱۳۵۰ می باشد همچنین فرمت وارد کردن زمان ها به صورت ۲۴ ساعت می باشد. در ترتیب وارد کردن روزهای هفته محدودیتی وجود ندارد در صورتکیه بخواهیم یک دستور در تمام روزهای هفته اجرا شود به جای وارد کردن نام تمامی روزها می توانیم از عبارت D به معنای تمام روزهای هفته استفاده کنیم.برای اینکه بتوانیم از Acl های برپایه ی زمان استفاده کنیم باید نام Acl زمانی تعریف شده را در انتهای Acl های معمولی ای که از قبل تعریف شده اند قرار دهیم. مثال های زیر به خوبی بیانگر چگونگی به کارگیری آنهاست:

```
acl time1 time SMTWHFA 07:00-12:00

acl time2 time SMTWHFA 12:01-19:00

acl time3 time SMTWHFA 19:01-23:59

acl office src 192.168.1.0/24

#####

http_access allow office time1

acl office2 src 10.0.0.1/24

http_access allow office2 time2

acl office3 src 10.0.0.1/24

http_access deny office2 time3
```

همانطور که می بینید Acl های برپایه ی زمان از قدرت فوق العاده ای به منظور کنترل هوشمندانه ی شبکه برخوردار هستند و به خوبی می توانند کاربران موجود در شبکه را در بازه های زمانی و روزهای مختلف هفته مدیریت و کنترل نمایند این روش در

سازمانهای بزرگ که کارکنان زیادی در بازه های زمانی مختلف مشغول به فعالیت هستند بسیار کاربردی است.به مثال بعدی توجه کنید:

```
acl morning_hrs time MTWHF 09:00-12:59
acl lunch_hrs time D 13:00-13:59
acl evening_hrs time MTWHF 14:00-18:00
acl youtube dstdomain .youtube.com
acl office dstdomain .office.example.com
######
http_access allow office
http_access allow youtube !morning_hours !evening_hours
http_access deny all
```

در اینجا کاربران موجود در ساعتهای کاری تعریف شده اجازه ی دسترسی به وب سایت تفریحی youtube را ندارند اما در ساعات استراحت میانه روز این کار مانعی ندارد در این مثال به جای تعریف دوباره زمانها علامت ! به کار رفته که به معنای انجام یک عمل خاص غیر از ساعات تعیین شده در acl های زمانی می باشد.

شناسایی URL ها

یکی دیگر از Acl هایی که برای کنترل محتوای وب به کار می رود url_regex است ،تفاوت این نوع با انواع قبلی دقت و سفارشی سازی بیشتر آن است به طور مثال در صورتیکه بخواهیم تمامی تصاویر با فرمت jpeg موجود در یک وب سایت را محدود کرده و از دسترسی کاربران به آنها جلوگیری کنیم، استفاده از این نوع مفید خواهد بود :

```
acl example_com_jpg url_regex ^http://example.com/.*\.jpg
http_access deny example_com_jpg
```

در این Acl تصاویر موجود در وب سایت example.com شناسایی و مسدود می شود مزیت این روش امکان اعمال کنترل بر جزئیات و گسترده بودن سطح عملکرد آن است.برای اینکه دستور فوق بتواند تمامی تصاویر موجود برروی سرور مورد نظر را

شناسایی و محدود نماید باید Acl تعریف شده را به صورت زیر تغییر دهیم:

```
acl example_com_jpg url_regex -i ^http://example.com/.*\.jpg
http_access deny example_com_jpg
```

پسوند i– موجب می شود تمامی داده های با فرمت jpg صرف نظر از کوچک و یا بزرگ بودن حروف شناسایی شوند .مشابه Aclدیگری به نام url_path_regex وجود دارد که هردو عملکرد مشابهی دارند تنها تفاوت آنها در این است که url_regex تمام رشته ی URL را برای یافتن عبارت مورد نظر جستجو می کند اما urlpath_regex تنهاURL را جستجو می کند به کارگیری این نوع زمانی مفید است که بخواهیم رشته ای از کلمات را جستجو کنیم به مثال زیر توجه کنید:

```
acl torrent urlpath_regex -i  torrent
```

همانطور که مشاهده می کنید در این نوع نیازی به وارد کردن آدرس دامنه نیست و Acl به طور خودکار موارد موردنظر را جستجو و شناسایی می کند.مثال زیر نیز برخی از فرمتهای رایج ویدویی را شناسایی می کند:

```
acl videos urlpath_regex -i \.(avi|mp4|mov|m4v|mkv|flv) (\?.*)?$
```

مطابقت دادن حساب های کاربری کاربران

سرویس دهنده اسکوئید قادر به استفاده از پروتکل ident به منظور شناسایی کاربران است . بدین منظور Acl نوع ident ایجاد شده است .برای این کار در ماشین کاربر مورد نظر باید پروتکل ident فعال شده باشد و نام کاربری کاربر نیز تعیین شده باشد در این حالت اسکوئید به ماشین کاربر توسط پروتکل ident متصل شده و نام کاربری موجود را شناسایی می کند همچنین نام کاربری تعیین شده در سمت کاربر با نام کاربری سمت پراکسی سرور ممکن است متفاوت.این حالت زمانی اتفاق می افتد که اسکوئید در تلاش برای اتصال به یک پراکسی سرور غیرفعال باشد دراین صورت نامهای کاربری squid، nobody، و یا proxy ممکن است دریافت شود. به مثال زیر توجه کنید:

```
acl friends ident john sarah michelle priya
http_access allow friends
http_access deny all
```

در اینجا اسکوئید به کاربرانی که دارای نام کاربری موجود در فهرست بالا باشند اجازه ی دسترسی به پراکسی سرور را خواهد داد.

*نکته: پروتکل **ident** از امنیت پایینی برخوردار است و هکرها به راحتی قادر به هک کردن نامهای کاربری و سوء استفاده*

از آنها هستند همچنین این پروتکل به دلیل بار پردازشی زیاد باعث ایجاد تاخیر در پاسخگویی می شود لذا در صورتکیه اجباری به استفاده از این پروتکل نیست از به کارگیری آن خودداری کنید.

معمولا تعریف نام های کاربری برای طیف وسیعی از کاربران غیر ممکن است جهت حل این مشکل دستور REQUIRED تعیین شده است این دستور یک نام کاربری را برای تمامی کاربران ایجاد می کند.

```
acl username ident REQUIRED
http_access allow username
http_access deny all
```

تعیین چارچوب نام های کاربری

مشابه دستور ident، Acl دیگری به نام ident_regex وجود دارد که محدوده ی ایجاد نام های کاربری را تعیین می کند این دستور زمانی که در شبکه سبک های مختلفی از نام های کاربری وجود دارد مفید است به طور مثال در یک سازمان برای هر بخش از کارکنان یک پیش وند تعیین شده است که بخش های مختلف از هم تمیز داده شوند در این صورت نام های کاربری ما به صورت زیر ساخته می شوند:

```
acl mkt_dept ident_regex -i \.marketing$
acl cust_care_dept ident_regex -i \.cust_care$
```

در این صورت نام های کاربری ساخته شده با پسوند های marketing و cust_care ایجاد می شوند و در واقع به نوعی محدوده ی نام های کاربری را تعیین کرده و از پراکندگی و بی نظمی آنها جلوگیری می کند.

احراز هویت در پراکسی سرور

بهترین راه برای جلوگیری از دسترسی افراد ی که به نوعی به دنبال خرابکاری در شبکه هستند احراز هویت توسط پراکسی سرور است ودر این حالت کاربران برای اینکه بتوانند به شبکه دسترسی داشته باشند باید نام کاربری و رمز عبور را از طریق مرورگر خود وارد نمایند.در صورتیکه سیستم احراز هویت فعال باشد هر کاربری که مرورگر خود را باز کند پیغامی مبنی بر وارد کردن نام کاربری و رمز عبور برای او فرستاده می شود و اسکوئید طبق طرح های از پیش تعیین شده تصمیم می گیرد که به کاربر درخواست کننده اجازه ی دسترسی به شبکه داده شود یا خیر. نکته ی جالب در این مورد این است که اسکوئید خود قادر به اعتبار سنجی نام های کاربری و کلمات عبور نیست و این پروسه از طریق یک فرایند پردازشی خارج از پراکسی سرور انجام می شود.بدین منظور از نوعی Acl به نام proxy_auth استفاده می شود که ما در آن مجموعه ای نام های کاربری مجاز را تعریف می کنیم همانطور که گفته شد اسکوئید خود قادر به تشخیص نام های کاربری نیست و ما باید حداقل یک طرح و رویه برای اعتبار سنجی نام های کاربری ایجاد نمائیم این فرایند از طریق دستور auth_param فایل پیکربندی ایجاد می

شود.اسکوئید به طور پیش فرض از ابزارهای مختلف احراز هویت مانند NTLM، Digests و Negotiate پشتیبانی می کند برای اجرای فرایند احراز هویت در اسکوئید باید خطوط زیر را به فایل پیکربندی اضافه کنیم:

```
acl authenticated proxy_auth REQUIRED
http_access allow authenticated
http_access deny all
```

با اضافه کردن خطوط بالا فقط کاربرانی که هنگام کاربری و رمز عبور آنها تایید شده باشد قادر به استفاده از پراکسی سرور می باشند اما در صورتکیه بخواهیم این فرایند را برای برخی از کاربران به صورت موردی طی شود و برای برخی از آنها مجوزهای دسترسی بالاتری را صادر کنیم باید نام های کاربری موردنظر را جداگانه تعریف کرده و مانند خطوط زیر به فایل پیکربندی اضافه نماییم:

```
acl authenticated proxy_auth REQUIRED
acl admins proxy_auth john sarah
acl special_website dstdomain admin.example.com
http_access allow admins special_website
http_access deny special_website
http_access allow authenticated
http_access deny all
```

در اینجا فقط نام های کاربری john و sarah اجازه دسترسی به محتوای admin.example.com را دارند و بقیه ی کاربران به تمامی وب ساتیها به جز admin.example.com اجازه ی دسترسی خواهند داشت.

تعیین ساختار نحوی نام های کاربری

مشابه proxy_auth دستور دیگری به نام proxt_auth_regex وجود دارد که از آن نیز به منظور تعیین نام های کاربری با یک فرمت مشخص به کار می رود در واقع مانند سایر Acl های قبلی که یک پیشوند برای تمامی نام های کاربری ایجاد می کردند این دستور نیز مشابه همین عمل را انجام می دهد .در مثال زیر تمامی نام های کاربری ایجاد شده همواره با پیشوند accounts_username ایجاد می شوند که به جای username نام کاربری مورد نظر قرار می گیرد که این کار در سازماندهی و مدیریت نام های کاربری در قالب های مختلف کاربرد زیادی دارد به مثال زیر توجه کنید:

```
acl accounts_dept proxy_auth_regex ^accounts_
```

در صورتکیه بخواهیم کاربران وجود در این محدوده ی نام های کاربری فقط به یک وب سایت دسترسی داشته باشند از دستورات زیر استفاده می کنیم:

```
acl accounts_dept proxy_auth_regex ^accounts_
```

```
acl accounts_web dstdomain .account.example.com
http_access allow accounts_dept accounts_web
http_access deny all
```

در این مثال کاربران موجود در بخش حسابداری سازمان فقط به وبگاه account.example.com دسترسی خواهند داشت.مباحث و دستورات موجود در رابطه با کنترل و احراز هویت در اسکوئید بسیار وسیع است تاکنون دستورات زیادی در این باره معرفی شدند در فصل هفتم نیز مباحث بیشتری در این باره مطرح خواهد شد.

محدودیت در تعداد اتصالات توسط یک کاربر

به طور کلی احراز هویت تعداد زیادی از کاربران نیازمند قدرت پردازشی زیادی است بنابراین در صورتکیه تعداد کاربران زیاد باشند و با توجه به اینکه منابع پردازشی همواره محدود هستند و همچنین سیستم عامل برای تعداد اتصلات همزمان محدودیت قائل می شود ،برقراری ارتباطات همزمان زیاد از طرف یک کاربر موجب کاهش راندمان پراکسی سرور و ایجاد وقفه در پردازش اطلاعات می شود به همین دلیل اسکوئید برای اتصالات موازی و همزمان از طرف یک کاربر مشخص محدودیت قائل شده است .Acl ی که بدین منظور ایجاد شده است maxconn نام دارد ، در صورتکیه یک کاربر با یک آدرس ip مشخص تعداد اتصالاتی را که با اسکوئید برقرار کرده از تعداد مشخصی که توسط خود ما تعیین می شود فراتر رود اجازه ی برقرار ارتباطات بیشتر از وی سلب خواهد کرد.مثال زیر برای هر کاربر ۲۵ اتصال مجاز تعیین کرده است:

```
acl connections maxconn 25
http_access deny connections
```

در این دستورات در صورتکیه کاربر بیش از ۲۵ بار در یک بازه ی زمانی اقدام به برقراری ارتباط با پراکسی سرور کند با پیغام عدم اجازه ی دسترسی یا Access denied مواجه خواهد شد.سناریوی دیگری که برای این بخش می توانیم متصور شویم این است که بتوانیم کاربران مختلف را گروه بندی کرده و برای هر رنج ip تعداد اتصلات مختلفی را در نظر بگیریم:

```
acl normal_users src 10.2.0.0/16
acl corporate_users src 10.1.0.0/16
acl norm_conn maxconn 15
acl corp_conn maxconn 30
http_access deny normal_users norm_conn
http_access deny corporate_users corp_conn
```

در دستورات بالا کاربران موجود در گروه normal_users می توانندحداکثر ۱۵ اتصال همزمان را برقرار کنند در حالیکه برای کاربران گروه corporate_users این تعداد ۳۰ اتصال تعیین شده است.

تعیین تعداد ورودهای مجاز توسط یک نام کاربری

یکی دیگر از Acl های مورداستفاده در این بخش max_user_ip است این دستور تعداد دفعات مجاز استفاده از یک نام کاربری را برروی بیش از یک ماشین مشخص می کند همچنین authenticate_ip_ttl میزان زمان تاخیر مجاز برای یک ip را تعیین می کند:

```
acl ip_limit max_user_ip 3
http_access deny ip_limit
```

در این مثال هر نام کاربری می توا ند برروی سه ماشین مختلف با آدرس های ip متفاوت مورد استفاده قرار گیرد .

نکته: قبل از به کارگیری این قسمت حتما باید یکی از ابزارهای شفیص هویت پیکربندی و فعال شده باشد.

شناسایی از طریق هدر های HTTP

درخواستها و پاسخ های مبادله شده میان کاربران و پراکسی سرور از طریق اطلاعات مخفی موجود در هدر های HTTP قابل شناسایی هستند.در ادامه به بررسی برخی از مهمترین این هدر ها می پردازیم.

هدر User-agent

تقریبا در تمامی درخواستهای HTTP هدر ی به نام User-Agent وجود که به طور عمده به منظور شناسایی نام و نسخه ی HTTP client به کار می رود.به طور مثال خطوط زیر این هدر برای نسخه ی 3.6 مروگر موزیلا به صورت زیر است:

```
Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9.2.6) Gecko/20100625
Firefox/3.6.6 GTB7.1
```

همانطور که مشاهده می کنید این اطلاعات نشان می دهد که نسخه ی 3.6.6 مرورگر موزیلا برروی سیستم عامل لینوکس 32 بیتی قرار دارد.

یکی از Acl های جالب توجه در این قسمت browser نام دارد که قادر است نوع مرورگرهای مورد استفاده ی کاربران را از طریق آنالیز اطلاعات موجود در هدر user-agent شناسایی کرده و محدودیتهایی را برای کار با این مرورگرها به وجود آورد.خطوط زیر تنظیمات پیکربندی این Acl را نشان می دهند:

```
acl allowed_clients browser -i firefox msie
http_access allow allowed_clients
http_access deny all
```

در اینجا فقط کاربران مجاز به استفاده از مرورگرهای موزیلا اینترنت اکسپلورر استفاده هستند و سایر مرورگها غیرمجاز تشخیص داده می شوند.

شناسایی از طریق هدر Referer

هر یک از درخواستهای HTTP هدر ی رشته ای به نام rererer را با خود حمل می کنند که وظیفه ی آن نگهداری نشانی کاربر اصلی درخواست کننده است در واقع از طریق اطلاعات موجود در این هدر می توان نشانی مرجع درخواست کننده را شناسایی کرد یک مثال برای هدر referrer به صورت http://www.google.com/search?rlz=1C1GGLS_enIN345IN345&sourceid=chrome&ie=UTF-8&q=what+my+user+agent می باشد.اسکوئید از طریق Acl نوع referer_regex این رشته از اطلاعات را با اطلاعات اصلی تطابق می دهد.مزیت عمده ی این کار جلوگری از آلوده شدن کاربران به انواع بدافزارهایی است که کاربران را به وب سایتهای حاوی کدهای مخرب هدایت می کنند.برای مثال:

```
acl malicious_website dstdomain .malicious.example.com
acl malicious_referer referer_regex -i malicious.example.com
http_access deny malicious_website
http_access deny malicious_referer
```

این تنظیمات پیکربندی از ورود کاربران به وب سایتهایی که هدف آنها نفوذ و خرابکاری است جلوگیری می کند.

شناسایی از طریق محتوای درخواستی

اسکوئید دو Acl به نام های req_mime_type و rep_mime_type به ترتیب جهت کنترل بر میزان آپلود و دانلود کاربرن در شبکه فراهم کرده است که هر دوی این دستورات از هدر های http جهت شناسایی محتوای مورد نظر استفاده می کنند به مثال زیر توجه کنید:

```
acl mpeg_upload req_mime_type -i video/mpeg
http_access deny mpeg_upload
```

این دستورات به کاربرن اجازه ی دانلود قایلهایی با فرمت mpeg را نخواهد داد.به طور مشابه rep_mime_type نیز جهت

جلوگیری از آپلود فایلهای با فرمت mpeg به کار می رود :

```
acl video_download rep_mime_type -i ^video/
http_reply_access deny video_download
```

نکته: عملکرد صحیح این **Acl** *ها وابسته به رعایت استانداردهای* **HTTP** *مانند تعیین هدر* **Content-Type** *از سوی وب سرورهای مقصد دارد.*

سایر هدر های HTTP

تاکنون با Acl های مختلفی چون browser, referer, req_mime_type و rep_mime_type آشنا شدیم که همگی از طریق به کارگیری هدر های مختلف HTTP منشا درخواستها و پاسخها را شناسایی می کنند.در همین رابطه دو ACl دیگر به نام های req_header و rep_header وجود دارند که مانند موارد قبلی وظیفه ی آنها نیز شناسایی انواع هدر های HTTP است. req_header به منظور شناسایی سرفایهای HTTP موجود در درخواستها به کار می رود.به مثال زیر توجه کنید:

```
acl user_agent req_header User-Agent -i ^Mozilla
http_access allow user_agent
http_access deny all
```

در این تنظیمات ، User-Agent یکی از انواع هدر های HTTP است به طور مشابه در صورتکیه بخواهیم اسکوئید هر یک از سرفایهای مختلف HTTP را با درخواستهای کاربران مطابقت دهد میتوان آن را در تنظیمات بالا اضافه نمود.به طور مشابه تنظیمات پیکربندی مربوط rep_header نیز مانند تنظیمات بالا است با این تفاوت که هدف این Acl ، شناسایی هدر های موجود در پاسخ هاست نکته ی مهمی که در این باره موجود دارد این است که استفاده از این Acl تنها زمانی مفید است که همراه با دستور http_reply_access به کار گرفته شوند.

آشنایی با پیامHTTP reply status

هنگامی که اسکوئید میان وب سرورهای مقصد و کاربران جهت دریافت درخواست ارسال شده از سوی آنها قرار می گیرد ، به ازای هر درخواست دریافت شده یک پاسخ از وب سرور مورد نظر دریافت می کند با توجه به توانایی وب سرورها در پاسخ دادن به درخواست مورد نظر پاسخ ارسال شده شامل کدهای بخصوصی است که هر کدام معنای متفاوتی دارند به طور مثال در صورتی که سرور مورد نظر قادر به پاسخ دادن به درخواست کاربر باشد ، کد پاسخ دریافت شده ۲۰۰ خواهد بود .

نکته: جهت آگاهی از فهرست کامل کدهای **HTTP Status** *به نشانی* **http://en.wikipedia.org/wiki/List_of_HTTP_status_codes** *مراجعه فرمائید.*

اسکوئید از Acl ی به نام http_status جهت شناسایی پاسخهای ارسال شده از سوی وب سرور ها از طریق کدهای برگشت داده شده از آنها استفاده می کند به طور مثال در صورتکیه بخواهیم تمامی کدهای خطاهای برگشت داده شده از سرورهای مقصد در محدوده ۵۰۰ تا ۵۱۰ را شناسایی کنیم ، تنظیمات پیکربندی آن به صورت زیر خواهد بود:

```
acl server_errors http_status 500-510
```

در اینجا میتوانید محدوده ی کدهای مورد نظر خود را وارد کنید.

شناسایی تصادفی درخواستها

Acl نوع random به منظور شناسایی تصادفی درخواستها با توجه به یک احتمال از پیش تعریف شده به کار می رود.کد زیر ساختار کلی این Acl را نشان می دهد:

```
acl ACL_NAME random probability
```

پارامتر probability به یکی از صورت های زیر قابل تعریف است:

- کسری: شکل به کارگیری کسری آن به طور مثال به صورت 3/2 یا 4/3 به کار می رود.
- اعشاری: شکل اعشاری آن به صورت 0.56 یا 0.5 یه کار می رود.
- خارج قسمت : شکل خارج قسمت آن به صورت 3:4 یا 2:3 می باشد.

مثال زیر 70 درصد از درخواستهای دریافت شده توسط اسکوئید را به صورت تصادفی با پاررامترهای مورد نظر ما مطابقت می دهد:

```
acl random_req random 0.7
```

آشنایی باAccess list rulesها

در بخش قبل مطالب جامعی در مورد ACL ها بیان شد. همانطور که مشاهده کردید این دستورات فقط برای شناسایی و تعیین خط مشی های کنترلی به کار می روند اما به خودی خود قادر به اعمال محدودیت و یا کنترل نیستند. به طور مثال ما توانستیم گروهی از کاربران با محدوده ی آدرسی مشخص را تعیین و شناسایی کنیم اما قادر به اعمال هیچ گونه کنترلی بر آنها نبودیم . به منظور اجرایی کردن سیاستهای تعیین شده در ACL ها ، از HTTP Access ها استفاده می کنیم . HTTP access ها قادر به

اعمال کنترل بر طیف وسیعی از ACL ها و یا فقط یک ACL مشخص هستند که از عبارت deny برای محدود کردن یک ACL و از Allow به منظور صدور اجازه ی دسترسی استفاده می شود.گستره ی عملکرد هریک از HTTP Access ها همانطورکه در فصل دوم مطرح شد میتواند بر یک Acl و یا فهرستی از Acl های هم نوع صورت گیرد.

دسترسی به پروتکل HTTP

http_access یکی از مهمترین access list های موجود به شمار می رود . هنگامی که کاربری اجازه ی دسترسی به این Rule را نداشته باشد اجازه ی دریافت و ارسال داده از طریق پروتکل HTTP را نخواهد داشت به همین خاطر نقش این Rule کلیدی تر از سایرین است. تنظیمات پیش فرض HTTP ACCESS تمامی درخواستها را به جزء کاربران موجود در گروه LAN مسدود می نماید و به هیچ یک از سایر کاربران ودرخواستها اجازه ی عبور از پراکسی سرور را نمی دهد دلیل اینکار حفظ امنیت پراکسی سرور قبل از پیکربندی آن عنوان شده است. همواره توصیه می شود کلاس های مختلف ip را در ACL ها جداگانه تعریف کرده و فقط به این آدرسها اجازه ی دسترسی داده شود و مابقی آدرسهای موجود توسط دستور http_access deny all مسدود شوند بدین ترتیب شبکه ی شما فقط از طریق افرادی که به نوعی در شبکه شما حضور دارند قابل دسترس خواهد شود این موضوع هر چند که ساده به نظر می آید اما در حفظ امنیت شبکه بسیار موثر است زیرا از یک طرف پهنای باند موجود توسط افراد متفرقه استفاده نمی شود و از طرف دیگر هویت کاربران ناشناخته باقی می ماند . به طور کلی همواره توصیه می شود پس از تعریف کلاسهای محختلف IP در ACL جداگانه و تعیین سطح دسترسی ها ، در پایان تمامی تعاریف یک ACL از نوع deny all تعریف شود.مثال زیر نحوه برخورد با Acl های مختلف را نشان می دهد:

```
http_access allow employees
http_access allow customers
http_access allow guests
http_access allow vpn_users
http_access deny all
```

همانطور که مشاهده می کنید ، در پایان تمامی تعاریف عبارت deny all آمده است.با توجه به سطح نیاز و گستردگی شبکه می توان Acl های مختلفی را تعریف کرده و سطح دسترسی آنها را مشابه موارد بالا تعیین کرد.

Adapted HTTP access

دستور adapted_http_access عملی مشابه http_access را انجام می دهد اما تفاوت آن در این است که رول های کنترلی ایجاد شده را پس از تمامی redirector ها اعمال می کند .این دستور فقط زمانی مفید است که از ابزاهای redirect همراه با اسکوئید استفاده شود همچنن طریقه ی ساخت این دستور کاملا مشابه با http_access است .

پیکربندی HTTP accessبرای پاسخ ها

Acl های مختلفی در قسمت های قبلی معرفی شدند که همگی آنها قادر به شناسایی نوع در خواستها و پاسخگویی به آنها بودند مانند src, dst, dstdomain, req_header rep_header, rep_mime_type . در صورتکیه بخواهیم سیاستهای کنترلی خود را برروی پاسخها هم پیاده سازی کنیم ، باید این Acl ها را همراه با http_reply_access به کار کار گرفته شوند. یادآوری این نکته ضروری است در صورتکیه یک کاربر و یا سرور مقصد از طریق http_access مسدودشده باشد تمامی درخواستهای ارسالی آن فرد و یا سرور در rule های reply_access نیز وضعیت مشابهی خواهند داشت و این دستورات نیز قادر به پاسخگویی به درخواستها ی مورد نظر نیستند .نحوه ساخت http_reply_access کاملا شبیه http_access است.

دسترسی به سایر درگاهها

پراکسی سرورهای همسایه می توانند از طریق درگاههای ICP و HTCP با پراکسی سرور شما ارتباط برقرار کنند همچنین پراکسی سرورها از طریق درگاه SNMP نیز قابل دسترسی هستند.در ادامه به چگونگی کنترل این درگاهها می پردازیم.

درگاه ICP

درگاه icp درگاهی است که اسکوئید از آن به منظور ارتباط با پراکسی سرورهای همسایه استفاده می کند.به منظور مدیریت و کنترل میزان و محدوده ی دسترسی به سرورهای همسایه رولی به نام icp_access ایجاد شده است.مقدار پیش فرض این عبارت deny all یعنی مسدود کردن تمامی ارتباطات است البته در یک شبکه ی محلی گزینه ی مناسب تر اجازه ی دسترسی به سرورهای محلی و کاربران شبکه جهت برقراری ارتباط و دسترسی موثر است اما در کل این موضوع به سیاست های شبکه شما بستگی دارد.جهت برقراری ارتباط با تمامی پراکسی سرورهای همسایه از دستورات زیر کمک می گیریم:

```
acl localnet src 10.0.0.0/8
acl localnet src 172.16.0.0/12
acl localnet src 192.168.0.0/16
acl localnet src fc00::/7
```

```
acl localnet src fe80::/10
icp_access allow localnet
icp_access deny all
```

درگاه HTCP

HTCP یا (hypertext caching protocol) جهت شناسایی و برقراری ارتباط با سایر کش سرورها استفاده می شود.نحوه عملکرد آن شبیه icp است.جهت برقراری ارتباط از دستور htcp_port استفاده می شود همچنین برای مدیریت و کنترل محدوده ی ارتباطی htcp_access به کار می رود به طور کلی نحوه رفتار و عملکرد آن شبیه icp_port است و اسکوئید نیز به طور پیش فرض تمامی درخواستهای htcp را مسدود می نماید.

کنترل درخواستهای پالایش از طریق HTCP

یک پراکسی سرور قادر به ارسال درخواستهای پالایش یا HTCP CLR به سوی سایر پراکسی سرورهای همسایه است.ممکن است بخواهیم دسترسی برخی از پراکسی سرورهای ناشناخته را محدود نمائیم.با استفاده از دستور ht cp_cl r _access میتوان کاربران و سرورهایی را که مایل به ارسال درخواست های پالایش داده ها نیستند را محدود کنیم.

درگاه SNMP

برای برقراری ارتباط از طریق درگاه snmp از دستور snmp_access استفاده می شود که جهت ایجاد محدودیت در ارتباط باید این دستور را همراه با نوعی acl به نام snmp_community به کار برد که مقادیر قابل استفاده در آن public یعنی سراسری و private به معنی محدودیت در ارتباط است.در مثال زیر چگونگی به کارگیری این Acl ها و همچنین چگونگی ایجاد محدودیت در ارتباطات به خوبی شرح داده شده است:

```
acl admins src 127.0.0.1 192.0.2.21 192.0.2.86
acl snmppublic snmp_community public
snmp_access allow snmppublic admins
snmp_access deny all
```

بدین ترتیب فقط مدیران شبکه از طریق آدرسهای 192.0.2.21 , 192.0.2.86 قادر به برقراری ارتباط از طریق درگاه snmp خواهند بود.توجه داشته باشید مقدار پیش فرض برای این درگاه نیز deny all یعنی مسدود کردن تمامی ارتباطات تعیین شده است.

اعمال محدودیت در برقراری ارتباط با پراکسی سرور های همسایه

هنگامی که پراکسی سرورهای یا کش سرور های همسایه در شبکه ما حضور داشته باشند هر یک از آنها می توانند به یکی از صورتهای parent و sibling با سرور شما در ارتباط باشند..زمانیکه این ارتباط به صورت Sibling باشد فقط فایلهایی که قبلا در مخزن کش سرور وجود داشته باشد و برچسب HIT داشته باشند مبادله می شود اما به طور عکس در صورتکیه ارتباط از نوع Parent باشد فقط درخواستهای با برچسب Miss مبادله خواهد شد.انتخاب هر کدام از این روشها به میزان پهنای باند شبکه و وضعیت سرور شما بستگی دارد.برای وادار کردن سایر پراکسی سرورها به اینکه از سرور شما به عنوان Sibling استفاده کنند باید دستور miss_access را در تنظیمات خود به کار ببریم.فرض کنیم در شبکه دو پراکسی سرور همسایه با آدرسهای 192.0.2.25 و 192.51.100.25 وجود دارند و میخواهیم با سرور اول ارتباط داشته باشیم و با دومی هیچ گونه ارتباطی برقرار نکنیم برای این کار دستورات زیر را به فایل پیکربندی اسکوئید اضافه می کنیم:

```
acl good_neighbour src 192.0.2.25
acl bad_neighbour src 198.51.100.25

miss_access allow good_neighbour # This line is not needed. Why?
miss_access deny bad_neighbour
miss_access allow all
```

در خط سوم در مقابل دستور توضیحی مبنی بر عدم نیاز به نوشتن این خط دستور وجود دارد دلیل آن این است که اسکوئید به طور پیش فرض تمامی درخواستهای MISS را با سایر پراکسی سرورهای همسایه ای که اجازه ی ارتباط با ما را دارند مبادله می کند تنها کاری که ما باید انجام دهیم این است که با سرورهای همسایه ای که مایل به برقراری ارتباط نیستیم سطح دسترسی را مسدود نمائیم که این کار از طریق دستور miss_access که در مثال بالا به کار رفته است امکانپذیر است.

درخواست ارتباط با پراکسی سرورهای همسایه

جهت برقراری ارتباط با کش سرور و پراکسی سرورهای همسایه دستور cache_peer ایجاد شده است این دستور نام ،نوع،آدرس ،شماره ی درگاه وسطح ارتباط سرور ها را تعیین می کند. سرورهای همسایه از طریق پروتکل های http,icp,htcp قادر به برقراری ارتباط با یکدیگر تحت شرایطی که ما تعیین می کنیم هستند که این عمل توسط cache_peer_access امکانپذیر است.جهت حصول اطمینان از ارتباطات و ایجاد انعطاف در آنها می توان فرامین cache_peer_access , cache_peer و ACL های مختلف را با هم ترکیب کرد.نحوه ایجاد ارتباط با یک سرور همسایه به طورکلی به شکل زیر است:

```
cache_peer_access CACHE_HOST allow allow|deny [!]ACL_NAME ...
```

برای درک این موضوع به مثال زیر توجه فرمائید:

```
cache_peer cache1.example.com parent 8080 3130 proxy-only weight=1
cache_peer cache2.example.com parent 3128 3130 proxy-only weight=2
acl youtube dstdomain .youtube.com
acl google dstdomain .google.com
cache_peer_access cache1.example.com allow youtube
cache_peer_access cache1.example.com deny all
cache_peer_access cache2.example.com allow google
cache_peer_access cache2.example.com deny all
```

دراین مثال دو cache peer به نامهای cache1.example.com و cache2.example.com وجود دارند که ارتباط این دو از نوع parent تعریف شده است و در ادامه شماره ی درگاههای ارتباطی میان این دو و همچنین سطح ارتباط آنها تعیین شده است.در خط سوم و چهارم دو acl به نامهای youtube و google تعریف شده است در خطوط بعدی توسط دستور cache_peer_access تمامی درخواستهایی که مرتبط با وب سایت یوتیوب است به سوی کش سرور اول و تمامی درخواستها مربوط به وب سایت گوگل نیز به سوی کش سرور دوم هدایت می شود و خط آخر بیان می کندکه هیچ ارتباط دیگری جز تبادل ترافیک این وب سایت ها میان این دو سرور انجام نخواهد شد.

ارجاع درخواست ها به سوی سرورهای مقصد

هنگامی که پراکسی سرورهای همسایه در قسمت cache_peer مشخص شدند اسکوئید قادر خواهد بود درخواستهای کاربران را جهت ایجاد تعامل و استفاده از منابع سایر کش سرورهای همسایه و بسته به نوع تنظیمات و ارتباط برقرار شده به آنها ارجاع دهد.گاهی اوقات به دلایلی تمایلی به انتقال و ارجاع تمامی درخواستها میان سرور های همسایه نداریم بدین منظور دو دستور always_direct و never_direct ایجاد شده اند و به ما امکان تعیین اینکه چه درخواستهایی از کاربران با سایر سرورها به اشتراک گذاشته شوند و یا درخواستها به طور مستقیم به سرورهای مقصد ارسال شوند.درصورتکیه بخواهیم درخواستها به طور مستقیم به سرورهای مقصد جهت پاسخگویی فرستاده شوند و به کش سرورهای همسایه منتقل نشوند باید از always_direct استفاده کنیم این دستور در مواقعی که مایل به اشتراک گذاری منابع موردنظر مثلا شبکه داخلی خود باسرورهای همسایه نیستیم استفاده می کنیم. مثال زیربه خوبی گویای این موضوع است:

```
acl local_domains dstdomain .local.example.com
acl local_ips dst 192.0.2.0/24
always_direct allow local_domains
always_direct allow local_ips
```

این دستورات به خوبی وقفه ی ایجاد شده در اثر ارتباط بی مورد با سرورهای همسایه را برطرف می کند زیرا به طور کلی سرعت انتقال اطلاعات در یک شبکه ی محلی بالا است و نیازی به اشتراک گذاری درخواستها نیست.

ACL نوع never_direct برعکس always_direct عمل می کند و تمامی درخواستها را قبل از پاسخگویی مستقیم توسط سرورهای مقصد با کش سرورهای همسایه مبادله می کند تا درصورتکیه پاسخ درخواست مورد نظر در آنها وجود داشته باشد از منابع آنها به جای مراجعه به سرورهای مقصد استفاده کند.شکل کلی دستور به صورت زیر است:

```
never_direct allow all
```

Ident lookup access

اگر مطالب بخش های قبل را به خاطر داشته باشید ACL ی به نام ident معرفی شد که وظیفه ی آن احراز هویت کاربران از طریق نام کاربری قبل از دسترسی مستقیم آنها به پراکسی سرور بود.به هر حال به طور پیش فرض احراز هورت کاربران از این طریق غیر فعال بوده و برای استفاده از این ویژگی باید از رول دسترسی نوع ident_lookup_access استفاده کنیم.از آنجا که تمامی میزبانهای موجود قادر به پشتیبانی از ویژگی ident نیستند می توانیم این قابلیت را برای گروهی مشخص از کاربران پیاده سازی کنیم.مثلا فرض کنیم می خواهیم این قابلیت را برای تمامی میزبانهایی که از سیستم عامل های لینوکس و یونیکس استفاده می کنند به کار بگیریم.برای این کار خطوط زیر را به فایل پیکربندی اضافه می کنیم:

```
acl nix_hosts src 192.0.2.0/24
ident_lookup_access allow nix_hosts
ident_lookup_access deny all
```

کنترل فرآیند کش کردن اسناد موجود در وب

اسکوئید به طور پیش فرض تمامی اطلاعات موجود در وب سایتها را به منظور نگهداری آنها جهت پاسخگویی به درخواستهای مشابه در مخزن کش خود نگهداری می کند که حاصل این کار افزایش رضایت کاربران در اثر افزایش سرعت دریافت اطلاعات مورد نظر خود است.اما در برخی موارد کش شدن همه ی اطلاعات از جمله محتوای موجود در شبکه ی محلی مورد نیاز نیست و یا به دلایلی مایل به نگهداری برخی از اطلاعات در مخزن اسکوئید نیستیم .به منظور کنترل فرایند کش کردن اطلاعات، ACL ی به نام cache ایجاد شده است که به کمک آن و سایر ACL ها می توانیم بر محتوای کش شده نظارت داشته باشیم.مثال زیر از کش شدن تمامی محتوای موجود در شبکه ی محلی جلوگیری می کند:

```
acl local_domain dstdomain .local.example.com
```

```
cache deny local_domain
cache allow all
```

سفارشی سازی صفحات خطا

هنگامی که کاربری اجازه ی دسترسی به کش سرور را پیدا نمی کند ،اسکوئید به طور پیش فرض با ارسال پیغام خطا در قالب صفحه ی وب به طور خلاصه دلایل ایجاد خطا را کاربران نمایش می دهد از طرفی می توان صفحات خطا را به یک آدرس URL مشخص ارجاع داد تا پیامی مشخص را به اطلاع کاربر برساند بهتر است طراحی صفحات طوری باشد که عوامل بروز این مشکل را به اطلاع کاربر رسانده و نشانی تماس با مدیر شبکه مثلا نشانی ایمیل وی در صفحه نمایش داده شود. deny_info دستوری است که بدین منظور ایجاد شده است و به طور کلی سه راه برای پیاده سازی این کار وجود: راه حل اول نمایش صفحه ی خطای تعریف شده در دایرکتوری خطاهای اسکوئید است:

```
deny_info ERR_PAGE ACL_NAME
```

در این حالت یک صفحه ی خطای نوشته شده به زبان HTML را در دایرکتوری خطاها نگهداری کرده و توسط دستور error_directory مسیر مورد نظر را در squid.conf تعریف می کنیم:

```
acl bad_guys src 192.0.2.0/24
deny_info ERR_CUSTOM_ACCESS_DENIED bad_guys
```

در اینجا فرض بر این است نام فایل نمایش دهنده صفحه ی خطا ERR_CUSTOM_ACCESS_DENIED است که قبلا در دایرکتوری خطاهای اسکوئید تعریف شده است .

راه حل بعدی نمایش یک صفحه ی وب به جای تعریف صفحه خطا است :

```
acl bad_guys src 192.0.2.0/24
deny_info http://errors.example.com/access_denied.html bad_guys
```

وب سایت مورد نظر به عنوان صفحه ی خطا در مواقعی که کاربر با خطایی مواجه می شود نمایش داده می شود بدیهی است طراحی این صفحه هم باید به گونه ای باشد که بیانگر وقوع یک خطا است.

راه حل آخر قطع ارتباط TCP کاربر است که باز هم صفحه ی خطای مرورگر به طور معمول نمایش داده می شود که آن هم بیانگر وقوع یک خطا در ارتباط است :

```
acl bad_guys src 192.0.2.0/24
deny_info TCP_RESET bad_guys
```

هریک از روش های معرفی شده به نوبه ی خود در هر موقعیتی می توان از آنها استفاده کرد.

ایجاد محدودیت در حجم داده های ارسالی

در بسیاری از موارد میزان پهنای باند موجود در شبکه محدود بوده و درصورتیکه مدیریت صحیحی بر دریافت داده ها صورت نگیرد شبکه با کمبود پهنای باندمواجه می شود. یکی از مهمترین عواملی که باعث مصرف بیش از حد پهنای باند می شود ،دریافت فایهای حجیم مانند فیلمها،بازیها است.برای حل این مشکل نوعی از ACL به نام reply_body_max_size ایجاد شده است که وظیفه ی آن ایجاد محدودیت در حجم داده های دریافتی توسط کاربران است و در صورتکیه حجم داده ارسالی از سوی پراکسی سرور از محدوده ی مجاز فراتر رود ،کاربر با پیغام خطا روبرو می شود .ساختار کلی دستور به شکل زیر است:

reply_body_max_size SIZE UNITS ACLNAME

در فایل پیکربندی زیر سقف مجاز برای دریافت داده از سوی کاربران که در محدوده ی آدرس متفاوت قرار دارند به ترتیب و ده و بیست مگابایت تعیین شده است:

```
acl max_size_10 src 192.0.2.0/24
acl max_size_20 src 198.51.100.0/24
reply_body_max_size 10 MB max_size_10
reply_body_max_size 20 MB max_size_20
reply_body_max_size none
```

کاربران موجود در گروه max_size_10 حداکثر حجم دریافتی یک فایل برای آنها 10 مگابایت و کاربران گروه acl max_size_20 حداکثر حجم یک فایل دریافتی 20 مگابایت تعیین شده است در صورتکیه در هرکدام از گروهها حجم داده دریافتی از محدوده ی تعیین شده فراتر رود کاربر مورد نظر با پیام خطا ی reply too large مواجه خواهد شد.مکانیسم انجام این کار توسط هدر connect-length انجام می شوددر صورتکیه در فایل دریافتی کاربر این هدر وجود نداشته باشد به جای نمایش پیغام خطا ارتباط کاربر با سرور مورد نظر قطع می شود.

ایجاد استثنا در ثبت درخواستهای کاربران

همانطور که قبلا گفته شد اسکوئید آدرس تمامی درخواستهای کاربران را در فایل ثبت رویداد access.log ثبت می کند.درصورتکیه بخواهیم در این امر استثنا قائل شویم از فرمان log_access استفاده می کنیم.این دستور که خود نوعی ACL

به شمار می رود ،هر آدرسی که در ACL همراه با آن تعریف شود ،درخواستهای فرستاده شده و دریافت شده توسط آدرس مورد نظر را در فایل ثبت رویداد ذخیره نمی کند .شاید بهترین دلیل برای انجام این کار حفظ حریم شخصی برخی از کاربران خاص است به هر حال نحوه انجام آن به صورت زیر است:

```
acl secret_req src 192.0.2.0/24
log_access deny secret_req
log_access allow all
```

در این مثال تمامی درخواستهای کاربران موجود در محدوده ی آدرس 192.0.2.0/24 در فایل ثبت رویداد access.log ذخیره نخواهد شد.در فصل پنج در این باره به طور مفصل بحث خواهیم کرد.

جلوگیری از کش شدن داده های موجود در شبکه محلی

هنگامی که اسکوئید به کش کردن اطلاعات می پردازد ،تمامی اطلاعاتی که قابل ذخیره سازی هستند را کش می کند اعم از دادهای موجود در شبکه ی محلی و یا داده های موجود در سایر وب سایتها،با توجه به اینکه همواره سرعت انتقال اطلاعات در شبکه های محلی بسیار بالاست ،نیازی به کش شدن اطلاعات جهت افزایش سرعت نیست و ذخیره سازی این اطلاعات نتیجه ای جز هدر رفتن حافظه ی دیسک سخت و حافظه ی موقت را در پی نخواهد داشت.برای جلوگیری از این کار در مرحله ی نخست باید درخواستهایی که مربوط به شبکه ی محلی هستند را از طریق آدرس های ip مبدا و مقصد شناسایی کنیم.در صورتکیه محدوده ی آدرس های موجوددر شبکه ی داخلی 192.0.2.0/24 و 198.51.100.0/24 باشند با ایجاد یک ACL می توان تمامی درخواستهای مربوط به شبکه ی داخلی را شناسایی کرد : acl client_servers dst 192.0.2.0/24 198.51.100.0/24 همچنین به طور مثال آدرس پست الکترونیکی ای که در شبکه ی داخلی یا اینترانت میزبانی می شود mail.internal.example.com است که با ساختن یک ACL می توان تمامی درخواستهای ارسال شده به آن را مشخص نمود: acl internal_websites dstdomain .internal.example.com بدین ترتیب می توان تمامی آدرس ها و سرورهای موجود در شبکه داخلی را توسط ACL ها شناسایی کرد .پس از طی این مرحله ،تنها کاری که باید انجام شود این است که اسکوئید اطلاعات مبادله شده میان آدرس های مشخص شده را ذخیره و کش نکند که این امر به سادگی امکانپذیر است:

```
cache deny client_servers

cache deny internal_websites

cache allow all
```

توسط این خطوط و دستورات قبل به راحتی می توان در منابع سخت افزاری اسکوئید صرفه جویی به عمل آورد.

کنترل بر نحوه بازنویسی آدرس های وب

هنگامی که همراه با اسکوئید از ابزارهای بازنویسی آدرس های وب (URL Rewriter) ها استفاده شود ، به طور پیش فرض اسکوئید تمامی درخواستهای ورودی را به سوی این ابزارها به منظور ایجاد تغییرات پردازش احتمالی سرازیر می کند که این موضوع به طور طبیعی باعث بالا سطح پردازش پردازنده می شود.برای اینکه بتوان سطح پردازش لازم را کنترل کرده و از مصرف بیش از اندازه و بیهوده ی توان پردازنده جلوگیری به عمل آورد ،میتوان اسکوئید را مجاب کرد تا فقط آدرسهای مشخصی را به سوی این ابزراها ارسال کرده و سایر آدرسها و درخواستها روند طبیعی خود را طی نمایند همچنین باید از بازنویسی درخواستهای نوع CONNECT، PUT و POSTجلوگیری کرد زیرا هرگونه تغییر در آدرس این نوع در خواستها موجب ایجاد رفتارهای غیر منتظره خواهد شد.با استفاده از دستورهای نوع url_rewrite_access میتوان فقط آدرسهای مشخصی را به سوی ابزارهای بازنویسی آدرسها ارسال کرد.عملکرد این دستور نیز مشابه رول های دسترسی نام آشنایی چون http_access و cache_peer_access است و با استافده از آن فقط آدرسهایی که در مقابل این دستور تعریف شده باشند به سوی ابزارهایی بازنویسی فرستاده می شوند.به طور مثال می خواهیم نشانی videos.example.com را به سوی ابزرهای بازنویسی ارسال کنیم برای این کار خطوط زیر را به فایل پیکربندی اسکوئید اضافه می کنیم:

```
acl video_web dstdomain .videos.example.com
url_rewrite_access allow video_web
url_rewrite_access deny all
```

مدیریت هدر های HTTP

یکی دیگر از دستوراتی که در این بخش مطرح می شود request_header_access و request_header_access نام دارند که وظیفه ی آنها اعمال کنترل و محدودیت بر هدر های HTTP است.این دستورات می توانند با سایر ACLها و رول های کنترلی موجود ترکیب شده و بر هدر های مختلف اعمال کنترل نمایند.به طور کلی همواره به یاد داشته باشید ، هرگونه تغییر و یا حذف هدر های HTTP باعث نقض استاندارهای پروتکل HTTPمی شود و ممکن است باعث ایجاد مشکلات غیر منتظره ای شود.به هر حال در مثال زیر می خواهیم هدر User-Agent را در درخواستها و پاسخهای موجود در محدوده ی آدرس 192.0.2.0/24حذف کنیم باید به صورت زیر عمل کنیم:

```
acl special_net src 192.0.2.0/24
request_header_access User-Agent deny special_net
reply_header_access Content-Type deny special_net
request_header_access User-Agent allow all
reply_header_access Content-Type allow all
```

جلوگیری از دسترسی کاربران شبکه خارجی به پراکسی سرور

یکی از اهداف مهم راه اندازی پراکسی سرور در شبکه ،جلوگیری از دسترسی کاربران ناشناس به شبکه و افزایش امنیت شبکه و کاربران است.برای اینکار کافی است کاربران و گروههای کاری موجود در شبکه داخلی را شناسایی کرده و از دسترسی سایر افراد به پراکسی سرور جلوگیری به عمل آوریم.خوشبختانه اسکوئید در مقوله ی امنیت به خاطر وجود ابزارهایی چون انواع ACL ها و access_rule ها بسیار قدرتمند و انعطاف پذیر است در طول فصل های گذشته با این ابزارها به طور مفصل آشنا شده ایم و در مواردی چون احراز هویت کاربران،مدیریت دسترسی به وب سایتها،جلوگیری از دسترسی به وب سایتها و موارد دیگر از آنها استفاده کردیم.اسکوئید در حالت پیش فرض خود نیز فقط به کاربران شبکه ی داخلی مجوز دسترسی را صادر کرده که این خود نشانگر امنیت بالای این سرویس دهنده است در ادامه نگاهی به فایل پیکربندی پیش فرض اسکوئید می اندازیم:

```
acl localhost src 127.0.0.1/32
acl localnet src 10.0.0.0/8
acl localnet src 172.16.0.0/12
acl localnet src 192.168.0.0/16
acl localnet src fc00::/7
acl localnet src fe80::/10
http_access allow localnet
http_access allow localhost
http_access deny all
```

همانطور که مشاهده می کنید ،این دستورات تمامی آدرس های ip ای که در شبکه ی داخلی به کار می روند را پوشش داده است و در صورتکیه بخواهیم به افرادی غیر از شبکه ی داخلی اجازه ی دسترسی به پراکسی سرور را بدهیم باید توسط یک acl

مجزا و صدور مجوز دسترسی از طریق http_access اجازه ی دسترسی را صادر کنیم.

جلوگیری از دسترسی گروه مشخصی ازکاربران

دلایل بسیار زیادی برای جلوگیری از دسترسی یک کاربر به شبکه وجود دارد که مهمترین آنها تلاش برای خرابکاری در شبکه مانند انتشار یک ویروس ،دانلود بیش از حد و یا ارسال داده های بسیار زیاد به سوی یک سرور مشخص که همگی آنها به نوعی باعث ایجاد اختلال در شبکه می شوند .برای جلوگیری از دسترسی این افراد مانند گذشته کافی است آدرس ip آنها را در یک ACL قرار داده و توسط access rule های موجود با آنها برخورد شود:

```
acl bad_clients src 192.0.2.21 198.51.100.25
http_access deny bad_clients
http_access allow localnet
http_access allow localhost
http_access deny all
```

جلوگیری از بارگذاری فایلهای تصویری

فایلهای ویدیوئی موجود به دلیل حجم بسیار بالای آنها هنگام بارگذاری حجم بسیار زیادی از پهنای باند را اشغال می کنند به طوریکه کاربران دیگر ممکن است قادر به باز کردن یک صفحه ی معمولی وب نباشند این در حالیست که پهنای باند موجود در شبکه می بایست در اختیار تمامی افراد قرار گیرد نه عده ی کمی از افراد.یکی از راههای جلوگیری از بروز این مشکل ممانعت از بارگذاری این فایلها توسط کاربران است.برای اینکه بتوانیم تمامی فرمت های تصویری را مسدود کنیم ابتدا باید تمامی این فرمتها را به اسکوئید معرفی کرده تا بتواند آنها را در محتوای مورد جستجوی کاربران شناسایی کند .برای این کار آسان ترین راه استفاده از ACL نوع url_regex به صورت زیر است:

```
acl video_content urlpath_regex -i \.(mpg|mpeg|avi|mov|flv|wmv|mkv|rm vb (\?.*)?$)
```

در برخی از وب سایتها که فرمت فایلهای موجود در سرور در انتهای url قرار نمی گیرد و به اصطلاح url های پویا نامیده می شوند وضعیت متفاوت است و acl بالا به تنهایی قادر به شناسایی تمامی فایلها نیست.به منظور کنترل بیشتر و شناسایی بهتر می توان از acl نوع rep_mime_type همراه با url_regex استفاده کرد:

```
acl video_in_reply rep_mime_type -i ^video\/
```

در مرحله ی بعد می بایست دسترسی به این فرمت را از طرف کاربران مسدود کنیم این کار از طریق http_access ها امکانپذیر است:

```
http_access deny video_content
http_reply_access deny video_in_reply
http_reply_access allow all
```

خط دوم از reply_access استفاده شده است این کار به این دلیل است در آدرس های پویا فرمتها قابل شناسایی نیستند و اسکوئید در پاسخی که به وب سرور مربوطه بر میگرداند اقدام به شناسایی محتوای مورد نظر می کند.بدین ترتیب هنگامی که کاربران بخواهند فایلهای تصویری را بارگذاری کنند با پیغام عدم دسترسی مواجه خواهند شد.

دسترسی ویژه برای برخی از کاربران

مدیران فنی برای دسترسی کامل به پراکسی سرور به منظور کنترل و مدیریت آن نیازمند دسترسی بالاتری نسبت به کارکنان معمولی هستند.به طور معمول اسکوئید بر تمامی کاربران سیاست یکسانی اعمال می کند در حالیکه مدیران شبکه خود نیازی به احراز هویت ندارند.برای رفع این محدودیت می توان به صورت زیر عمل کرد:

```
acl admin_laptops src 192.0.2.46 192.0.2.9 192.0.2.182
acl authenticated proxy_auth REQUIRED
acl admin_user proxy_auth john michelle sarah
acl work_related_websites dstdomain "/opt/squid/etc/work_websites"
http_access allow admin_laptops
http_access allow admin_user
http_access allow localnet work_related_websites authenticated
http_access deny all
```

اسامی مدیران تعیین شده sarah, john و michelle هستند این افراد با وارد کردن نام کاربری خود می توانند به آدرسهای ویژه ای که در فهرست آدرس های موجود در فایل /opt/squid/etc/work_websites دسترسی داشته باشند در حالیکه سایر کاربران اجازه ی چنین کاری را ندارند همچنین آدرس های ip این افراد نیز در یک ACL جداگانه تعریف شده در خط

آخر دسترسی سایر افراد به این آدرس ها مسدود شده است.در اینجا مدیران از طریق آدرس ip و نام کاربری احراز هویت شده اند و از هرکدام از روشها قادر به مشاهده این آدرسهای ویژه هستند.

ایجاد محدودیت در ساعات کاری

در برخی سازمانها به کارکنان خوددر برخی از ساعات استراحت دسترسی به محتوای غیر کاری مانند وب سایتهای سرگرمی ،بازی و... را مجاز می دانند برای اینکه بتوانیم به طور هوشمند این کار را انجام دهیم و نیازی به تغییر روزانه آن نباشیم می توانیم از ACL های بر پایه ی زمان استفاده کنیم :

```
acl working_hours time D 10:00-13:00

acl working_hours time D 14:00-18:00

http_access allow working_hours localnet work_related

http_access allow !working_hours localnet

http_access deny all
```

ساعات تعیین شده برای استراحت در این سازمان 13 تا 14 است.معمولا در یک سازمان در طول ساعات کاری فقط وب سایتهای مشخصی بازدید می شوند ،می توان آدرس تمامی این وب سایتها را در یک فایل متنی مثلا به نام work_related.txt ذخیره کنیم وACL مود نظر را به صورت acl work_related dstdomain ""/opt/squid/etc/work_related.txt بنویسیم و مطابق دستورات بالا کارکنان در ساعات کاری تعیین شده فقط می توانند از وب سایتهایی که در work_related.txt تعریف شده بازدید کنند ولی در زمان استراحت می توانند به تمامی وب سایتها دسترسی داشته باشند .

دسترسی کاربران ویژه به درگاههای مخصوص

در بخش های قبل در مورد درگاهها و چگونگی ایجاد و اعمال محدودیت در آنها مطالب مفصلی مطرح شد.برخی از درگاهها مانند 80 که ترافیک http از طریق آن منتقل می شود عمومی هستند و تمامی کاربران جهت تبادل اطلاعات از آن استفاده می کنند اما برخی از درگاهها فقط برای برقراری ارتباط میان دو مرکز ایجاد شده اند و کاربران عادی از وجود آن بی اطلاع هستند.برا ی ایجاد این درگاهها می توان آنها را در کنار سایر درگاهها در safe_ports اضافه نمود و یا می توان یک ACL جداگانه تعریف کرد:

acl special_ports port 119 2082 3389 9418

از آنجا که این درگاهها به صورت خصوصی ایجاد می شوند و سایر کاربران نباید به آن دسترسی داشته باشند باید از احراز هویت استفاده کردتا کاربران دیگر را از دسترسی به این درگاهها دور نگه داریم.احراز هویت کاربران از طریق نام های کاربری یا آدرس های ip آنها امکان پذیر است:

acl special_clients src 192.0.2.9 192.0.2.46 192.0.2.182

acl authenticated proxy_auth REQUIRED

acl special_users proxy_auth sarah john michelle

http_access allow special_ports special_clients

http_access allow special_ports special_user

http_access deny !Safe_ports

در اینجا باز هم از هر دو روش نام های کاربری و ip استفاده شده است در خط اول آدرس ip کاربران مورد نظر تعریف شده است ،در خطوط بعدی نام های کاربری به همراه مجوز دسترسی به آنها تعیین شده است و خط آخر نیز دسترسی به درگاههای غیر مجاز مسدود شده است.

خلاصه فصل

ACL ها و Access Rule ها از مهمترین دستورات موجود در پیکربندی اسکوئید به شمار می روند به طوریکه بدون وجود آنها اسکوئید قادر به هیچگونه کنترلی بر کاربران و منابع نیست.در این فصل با روش های مختلف تعریف،به کارگیری،اعمال این دستورات و ترکیب آنها آشنا شدیم همچنین چگونگی کنترل بر منابع شبکه،ایجاد محدودیت در مشاهده وب سایتها،اعمال کنترل بر محتوای بارگذاری شده توسط کاربران به طور مفصل بحث شد.به طور کلی مطالب مطرح شده در این فصل شامل موارد زیر است:

- معرفی انواع ACL ها
- تعریف ACL ها و به کارگیری آنها در کنار هم
- معرفی انواع Access Rule ها
- چگونگی تعریف Access Rule ها و به کارگیری آنها در کنار Acl ها
- استفاده از نام های کاربری و آدرس های ip جهت شناسایی و احراز هویت کاربران
- استفاده از Acl های بر پایه زمان به منظور کنترل کاملا پویا بر شبکه و کاربران

در این فصل به دلیل اهمیت بسیار زیاد این دستورات مثال های متنوعی از سناریوهای مختلف مطرح شد تا کاربران بیش از پیش با عملکرد این دستورات آشنا شوند .

فصل پنجم

آشنایی با فایلهای ثبت رویداد

به طور کلی درک عملکرد و ساختار فایلهای ثبت رویداد اسکوئید بسیار ساده است.ما در این فصل به معرفی فایلهای ثبت رویداد و چگونگی پیکربندی و سفارشی سازی آن می پردازیم.در این فصل همچنین چگونگی پیکریندی صحیح فایلهای ثبت رویداد به منظور حفظ حریم خصوصی کاربران ، معرفی انواع فایلهای ثبت رویداد ، درک تفاوت آنهاوهمچنین پیکربندی صحیح آنها متناسب با سیاست های موجود در شبکه به طور مفصل معرفی و بحث خواهند شد.

به طور کلی در این فصل در مورد عناوین زیر بحث خواهد شد:

- Cache log
- Access log
- سفارشی سازی access log
- چگونگی حفظ حریم خصوصی کاربران
- Referer log
- User Agent log
- گرد کردن logها

آشنایی پیام های رویداد

پیام های رویداد پیام هایی هستند که سیستم عاملها و بسیاری از نرم افزارها از آن برای ثبت فعالیت ها ی میان ماشین و افراد استفاده می کنند.این پیام ها خطاها،مشکلات افزاری و نرم افزاری و حتی فعالیت های کاربر را گزارش می کنند و یکی از بهترین روش های اشکال یابی است. اسکوئید نیز به دلیل نقش حساس آن در شبکه تمامی آنچه را که رخ می دهد را ثبت می کند این گزارشات مشکلات پیش آمده در اسکوئید،کاربران و ارتباطات شبکه را شامل می شود.همچنین به منظور دسترسی آسان و جلوگیری از سردرگمی پیام های مختلف را در فایلهای جداگانه ای ذخیره می نماید.cache.log نام فایلی است که اسکوئید از آن به منظور ثبت مشکلات ناشی از خطاهای پیکربندی،مشکلات ناشی از کش سرور ،کمبود منابع سخت افزاری را در آن ثبت و ذخیره می کند.همچنین تمامی فعالیت های کاربران در شبکه مانند دریافت و ارسال درخواستها در فایلی جداگانه به نام access.log ذخیره می شود.این فایل درخواستهای ارسالی از کاربران به همراه حالت پاسخ گویی اسکوئید به آن را ثبت می نماید که یکی از منابع مهم برای آنالیز عملکرد اسکوئید و میزان موفقیت آن در صرفه جویی از منابع شبکه به شمار می رود.

آشنایی با Cache log

اسکوئید،از زمان آغاز فعالیت تا زمانیکه به هر دلیلی از فعالیت بازمی ایستد تمامی رویدادهای ممکن را در cache.log ثبت می کند البته تمامی این گزارشات مربوط به اشکالات پیش آمده نیست چه بسا در طول یک دوره ی کاری هچ گونه خطایی گزارش نشود بلکه رویدادهایی چون اضافه شدن دایرکتوری های جدید،بررسی مخزن کش، زمان راه اندازی و توقف کش سرور و بسیاری از موارد دیگر را ثبت و ذخیره می کند تا در مواقع ضروری راه گشای مشکلات باشد.خطوط زیر نمونه ای گزارشاتی است که در cache.log ثبت می شود:

2010/09/10 23:31:10| Starting Squid Cache version 3.1.10 for i686-pc-linux-gnu...

2010/09/10 23:31:10| Process ID 14892

با دقت در خطوط گزارش متوجه می شوید که در خط اول نسخه ی اسکوئید مورد استفاده به همراه تاریخ و زمان راه اندازی نمایش داده می شود.همچنین در خط دوم شناسه ی ID اسکوئید که توسط سیستم عامل شناسایی می شود را نشان می دهد.

هنگامی که اسکوئید راه اندازی می شود ،DNS های درنظرگرفته شده در فایل پیکربندی بررسی می شوند و جزئیات آن را در cache.log ذخیره می کند:

2010/09/10 23:31:10| Initializing IP Cache...

2010/09/10 23:31:10| DNS Socket created at [::], FD 7

2010/09/10 23:31:10| Adding nameserver 192.0.2.86 from /etc/resolv.conf

اسکوئید همچنین ویژگیهای فعال و یا غیر فعال شده در هنگام کامپایل را از حیث فعال و یا غیر فعال بودن بررسی می کند:

2010/09/10 23:31:10| User-Agent logging is disabled.

2010/09/10 23:31:10| Referer logging is disabled.

فایلهای ثبت رخداد نیز در راه اندازی مجدد از جهت نام و مکان بررسی میشوند:

2010/09/10 23:31:10| Logfile: opening log daemon:/opt/squid/var/logs/

access.log

2010/09/10 23:31:10| Unlinkd pipe opened on FD 13

2010/09/10 23:31:10| Local cache digest enabled; rebuild/rewrite every

3600/3600 sec

2010/09/10 23:31:10| Store logging disabled

از مهمترین پیام هایی که در cache.log ثبت می شود،گزارشات مربوط به وضعیت دایرکتوریهای کش به همراه میزان فضای باقی مانده ومسیر دایرکتوریها را نشان می دهد:

2010/09/10 23:31:10| Swap maxSize 1024000 + 262144 KB, estimated 98934

objects

2010/09/10 23:31:10| Target number of buckets: 4946

2010/09/10 23:31:10| Using 8192 Store buckets

2010/09/10 23:31:10| Max Mem size: 262144 KB

2010/09/10 23:31:10| Max Swap size: 1024000 KB

2010/09/10 23:31:10| Version 1 of swap file with LFS support

detected...

2010/09/10 23:31:10| Rebuilding storage in /opt/squid/var/cache

(DIRTY)

2010/09/10 23:31:10| Using Least Load store dir selection

2010/09/10 23:31:10| Set Current Directory to /opt/squid/var/cache

واژه ی swap که در این پیام ها به کار رفته مربوط به دایرکتوریهای کش سرور است و با واژه ی swap که در سیستم عامل به کار می رود کاملا متفاوت است.

اسکوئید تمامی ماژولهای موردنیاز برای دریافت درخواست از کاربران را راه اندازی می کند:

2010/09/10 23:31:10| Loaded Icons.

2010/09/10 23:31:10| Accepting HTTP connections at [::]:3128, FD 16.

2010/09/10 23:31:10| HTCP Disabled.

پس از هر بار راه اندازی ، تمامی دایرکتوریهای موجود برروی دیسک سخت بررسی می شوند و خطاهای احتمالی گزارش می شوند:

2010/09/10 23:31:10| Done reading /opt/squid/var/cache swap

entries)

2010/09/10 23:31:10| Finished rebuilding storage from disk.

2010/09/10 23:31:10| 0 Entries scanned

2010/09/10 23:31:10| 0 Invalid entries.

2010/09/10 23:31:10| 0 With invalid flags.

2010/09/10 23:31:10| 0 Objects loaded.

2010/09/10 23:31:10| 0 Objects expired.

2010/09/10 23:31:10| 0 Objects cancelled.

2010/09/10 23:31:10| 0 Duplicate URLs purged.

2010/09/10 23:31:10| 0 Swapfile clashes avoided.

2010/09/10 23:31:10| Took 0.03 seconds (0.00 objects/se

2010/09/10 23:31:10| Beginning Validation Procedure

2010/09/10 23:31:10| Completed Validation Procedure

```
2010/09/10 23:31:10|   Validated 25 Entries
2010/09/10 23:31:10|   store_swap_size = 0
2010/09/10 23:31:11| storeLateRelease: released 0 objects
```

همه ی پیام هایی که در بخش قبل مطرح شدند مربوط به زمانی است که اسکوئید بدون هیچ گونه مشکلی راه اندازی می شود در واقع در گزاشات بالا هیچ گونه پیامی مبنی بر وجود مشکل دیده نمی شود.

اگر هنگام راه اندازی مجدد اسکوئید مجوزهای لازم برای نوشتن در فایلهای ثبت رویداد و یا ذخیره داده ها در دایرکتوریها را نداشته باشد با پیام های زیر مواجه می شوید:

```
2010/09/10 01:42:30| Max Mem  size: 262144 KB
2010/09/10 01:42:30| Max Swap size: 1024000 KB
2010/09/10 01:42:30| /opt/squid/var/cache/00: (13) Permission denied
FATAL:  Failed to verify one of the swap directories, Check cache.log
  for details.  Run 'squid -z' to create swap directories
  if needed, or if running Squid for the first time.
Squid Cache (Version 3.1.10): Terminated abnormally.
```

هنگام بروز خطا ،پس از گزارش علل مشکلات پیش آمده ،راه حل ممکن برای رفع مشکل نیز در ادامه ی گزارش نمایش داده می شود.

Access Log

آنچه که درcache.log ثبت می شود مربوط به فعالیت اسکوئید ، وضعیت منابع و گزارش خطاهای احتمالی است اما همانطور که می دانیم اسکوئید یک وظیفه ی مهم دیگر هم دارد و آن هم گزارش گیری از کاربران و ترافیک شبکه است که این کار توسط access.log انجام می شود در واقع access.log تمامی فعالیتهای کاربران و کسانی که پراکسی سرور دسترسی داشته باشند را ثبت می کند همچنین وضعیت پاسخگویی اسکوئید به یک درخواست مشخص و زمان ایجاد یک درخواست را نشان می دهد.مسیر پیش فرض access.log معمولا در ${ prefix}/var/logs/access.log قرار دارد که prefix بسته به مسیر و نحوه نصب و مسیردهی به آن متفاوت است.

آشنایی با access log

پیام هایی که در access.log ذخیره می شوند به طورکلی با آنچه که در cache.log دیدیم متفاوت اند در این فایل خبری از گزارش گیری از فعالیت اسکوئید و پیام های خطا نیست.در این فایل فعالیت های کاربران در شبکه و همچنین اطلاعات مبادله شده به همراه وضعیت دریافت و پاسخ درخواستها گزارش می شود .

به مثال زیر توجه کنید:

1284565351.509 114 127.0.0.1 TCP_MISS/302 781 GET http://www.

google.com/ - FIRST_UP_PARENT/proxy.example.com text/html

1284565351.633 108 127.0.0.1 TCP_MISS/200 6526 GET http://www.

google.co.in/ - FIRST_UP_PARENT/proxy.example.com text/html

1284565352.610 517 127.0.0.1 TCP_MISS/200 29963 GET http://www.google.co.in/images/srpr/nav_logo14.png - FIRST_UP_PARENT/proxy.example.com image/png

1284565354.102 147 127.0.0.1 TCP_MISS/200 1786 GET http://www.

google.co.in/favicon.ico - FIRST_UP_PARENT/proxy.example.com image/x-icon

در این خطوط زمان وقوع این گزارشات توسط زمان یونیکسی (unix time) نمایش داده می شوند در قسمت های بعدی آدرس ip دریافت کننده ی درخواست و ارسال کننده ی آن قرار می گیرد و در بخش های بعدی نیز پروتکل مورد استفاده و وضعیت پاسخگویی به درخواست و بالاخره آدرس های وب نمایش داده می شوند .ازآنجا که زمان یونیکسی برای بسیاری از کاربران نامفهوم است با استفاده از یک اسکریپت ساده می توان این زمان را به زمان استاندارد موجود تبدیل کرد:

```
$ perl -p -e 's/^([0-9]*)/"["localtime($1)."]"/e' < access.log > access.log.h
```

با وارد کردن این اسکریپت گزارشات بالا به صورت زیر نمایش داده می شوند:

[Wed Sep 15 21:12:31 2010].509 114 127.0.0.1 TCP_MISS/302 781 GET

http://www.google.com/ - FIRST_UP_PARENT/proxy.example.com text/html

[Wed Sep 15 21:12:31 2010].633 108 127.0.0.1 TCP_MISS/200 6526 GET

http://www.google.co.in/ - FIRST_UP_PARENT/proxy.example.com text/html

[Wed Sep 15 21:12:32 2010].610 517 127.0.0.1 TCP_MISS/200 29963 GET

http://www.google.co.in/images/srpr/nav_logo14.png - FIRST_UP_PARENT/

proxy.example.com image/png

[Wed Sep 15 21:12:34 2010].102 147 127.0.0.1 TCP_MISS/200 1786 GET

http://www.google.co.in/favicon.ico - FIRST_UP_PARENT/proxy.example.com

image/x-icon

همانطور که واضح است زمان یونیکسی به فرمت زمان فعلی تغییر پیدا کرده و زمان ایجاد هرکدام از گزارشات به طور دقیق نمایش داده می شوند.همچنین نوع متد http مورد استفاده در هرکدام از درخواست ها نیز گزارش می شوند که در این مثال متد GET به کاررفته است. همانطور که می دانیم متدهای مختلفی وجود دارند از جمله DELETE, PUT و POST که بسته به نوع داده مبادله شده متفاوت است که البته تفاوت آنها قبلا ذکر شده است.گزارشات ایجاد شده توسط اسکوئید به بسیاری از سوالات مدیران شبکه پاسخ می دهد و با بررسی دقیق آن با استفاده از نرم افزارهای آنالیز فایلهای ثبت رخداد اسکوئید می توان به اطلاعات ارزشمندی از شبکه و علاقه مندی کاربران و همچنین فعالیتهای آنان دست یافت.

ساختار Access log

می توان مکانهای مختلفی را برای ذخیره سازی فایلهای ثبت رویداد access log تعیین کرد همچنین با استفادهاز دو دستور access_log و logformat می توان مکان و فرمت تهیه ی ذخیره سازی گزارشات را تعیین نمود.به طور کلی ساختار نحوی access_log به صورت زیر است:

access_log <module> : <place> [<logformat name> [acl acl ...]]

در فیلد ماژول یکی از متدهای stdio ، none ، syslog، tcpو یا udpقرار می گیرد که سبک کلی تهیه ی گزارشات و ذخیره ی آنها در فیلد place را تعیین می کنند.در ادامه به بررسی هرکدام از این متدها می پردازیم.

- none: رخداد ها هرگز ثبت نخواهند شد.
- stdio: رخداد ها پس از کامل شدن هر درخواست سریعا ثبت می شوند.
- daemon:این ماژول نیز عملکردی مشابه stdio دارد اما پیام ها را برروی دیسک سخت ذخیره نمی کند.
- tcp: در صورت استفاده از ماژول tcp، رخدادها به سوی یک دریافت کننده ی مجهز به پروتکل TCPارسال می شود.نحوه تعیین مسیر دریافت کننده ی پیام ها به صورت host:port \\ می باشد.
- Syslog: این ماژول تمامی پیام های ثبت رویداد را با استفاده از امکانات ثبت رویداد سیستم عامل یعنی syslog ثبت و ذخیره می کند.پارامتر place نیز قالب کلی گزارش دهی برپایه ی syslogرا مشخص می کند که مقادیر مورد استفاده ی مجاز همراه با syslogشامل authpriv،daemon،local0 ، local1local7 وuser می

باشد.همچنین مقادیر مجاز برای استفاده جهت تعیین دوره ی ثبت پیام ها err،warning،notice ، info ، and debug می باشند.

- udp: مشابه ماژول tcp، رخداد ها به سوی یک دریافت کننده از نوع پروتکل UDPارسال می شود.نحوه تعیین مسیر دریافت کننده به صورت \\host:port می باشد.

همچنین با استفاده از دستورهای بالا می توان یک نام دلخواه برای logformat تعیین کرد و با استفده از ACLمختلف کنترل بیشتر و بهتری را بر فرایند ثبت گزارشات اعمال کرد.ساختار و پیکربندی پیش فرض access logدر اسکوئید به صورت زیر است:

```
access_log daemon:/opt/squid/var/logs/access.log squid
```

در این تنظیمات /opt/squid/به عنوان پیشوند پیش فرض و squidبه عنوان نام پیش فرض logfromat تعیین شده است.

قالب کلی Log ها

در بخش قبل چگونگی ارسال فایلهای ثبت رویداد به مکانهای مختلف را آموختیم حال قالب کلی این پیام ها و چگونگی سفارشی سازی آنها را می آموزیم.

قالب کلی پیام های رخداد از طریق دستور logformat تعیین می شود.ساختار کلی تعریف logformat به صورت زیر است:

```
logformat <name> <format specification>
```

format specification شامل گروهی از کدهای مخصوص قالب بندی است که توضیحا هر یک در جدول زیر آمده است:

قالب کد	توضیحات
%	یک کاراکتر حرفی (لیترال) محسوب می شود.
sn	یک شماره ی ترتیبی منحصر به فرد است که برای تعیین خطوط پیام ها به کار می رود.
err_code	شماره ی شناسایی کد های خطای داخلی تعیین شده برای اسکوئید.
err_detail	توضیحاتی را در مورد کدهای خطای موجود در err_code .
>a	تعیین آدرس IPکاربران .
>A	نام کامل و دقیق دامنه (FQDN) .

>p	منبع درگاه کاربران .
<A	آدرس IP و یا نام سرورهای همسایه .
la	آدرس IPمحلی مربوط به پراکسی سرور اسکوئید .
lp	درگاه محلی ای که اسکوئید به آن گوش می کند .
<lp	تعیین کننده ی آخرین درگاه به کاررفته توسط یک سرور و یا ارتباط .
ts	این کد بیانگر ثانیه در ساعتهای یونیکسی به کاررفته در پیام های رخداد می باشد.
tu	بیانگر میلی ثانیه ها در ساعت یونیکسی است.
tl	آرگومان اختیاری strftime مربوط به ساعت محلی را نشان می کند که قالب پیش فرض آن به صورت %d/%b/%Y:%H:%M:%S %z می باشد.
tg	این آرگومان نیز اختیاری بوده و ساعت را به وقت گرینویچ GMTبر می گرداند.مقدار پیش گزیده ی آن %d/%b/%Y:%H:%M:%S %z می باشد.
tr	زمان پاسخ به صورت میلی ثانیه را نشان می کند.
dt	زمان لازم برای جستجوی DNSها به میلی ثانیه .
[http::]>h	این آرگومان اختیاری نام اصلی هدر HTTP برای درخواستها به صورت [: [separator]element] .
[http::]>ha	هدر های HTTP موجود در درخواستها را پس از بازنویسی و تغییر نشانی تعیین می کند.مقدار پیش فرض برای این هدر ها h< می باشد.
[http::]un	نام کاربری را مشخص می کند.
[http::]<h	هدر موجود در پاسخ ها را برمی گرداند.مقدار پیش فرض برای این آرگومان h< می باشد.
[http::]ul	نام های کاربری مورد استفاده در احراز هویت را نشان می دهد.
[http::]ui	نام های کاربری مورد استفاده در درخواستهای identرا نشان می دهد.
[http::]>Hs	کد ارسال شده ی نشان دهنده وضعیت پروتکل HTTP به سوی کاربر را نشان می دهد.
[http::]<Hs	کد وضعیت HTTP دریافت شده را نشان می دهد.
[http::]<bs	تعداد پیام های معادل HTTP که توسط کاربر در اتصالات دریافت می شود به استثنای کدهای یونیکد را نشان می دهد.
[http::]Ss	وضعیت درخواستهای اسکوئید مانند TCP_MISSو ...
[http::]Sh	وضعیت اتصال اسکوئید به سایر پراکسی سرورها شامل DEFAULT_PARENT و...
[http::]mt	محتوای مربوط به کنترل MIME در پاسخها
[http::]rm	متدهای درخواست HTTPاز جمله POST ، GET و ... را نشان می دهد.
[http::]ru	درخواست URLرا نشان می دهد.

مسیر درخواست URLبه استثنای نام میزبان را نشان می دهد.	[http::]rp
درخواست نسخه ی پروتکل را برمی گرداند.	[http::]rv
ارسال حجم پاسخ ارسالی شامل هدر های HTTP	[http::]<st
دریافت حجم درخواست شامل هدر های HTTP به استثنای سایر کدها مانند یونیکدها	[http::]>st
اندازه ی هدر های موجود در درخواست ها را نشان می دهد.	[http::]>sh
اندازه ی هدر های موجود در پاسخ ها را نشان می دهد.	[http::]>sh
حجم درخواست و پاسخ ها و همچنین هدر های HTTPرا برمی گرداند.	[http::]st
پاسخ offset بالا فرستاده شده را نشان می دهد.	[http::]<sH
میزان فعلی حجم یک شیء را برمی گرداند.	[http::]<sS
زمان پاسخگویی کش سرور همسایه به میلی ثانیه.شمارنده از لحظه ی ارسال درخواست به سوی سایر همسایه ها تا لحظه ی دریافت آخرین بایت از پاسخ .	[http::]<pt
زمان کلی تبادل داده با سرور همسایه براساس میلی ثانیه. شمارنده از لحظه ی آغاز عملیات خواندن و نوشتن ادامه می یابد.	[http::]<tt

قالب های گزارش گیری فراهم شده توسط اسکوئید

به طور پیش فرض ، اسکوئید ۴ قالب کلی گزارش گیری را به صورت زیر فراهم کرده است:

```
logformat squid %ts.%03tu %6tr %>a %Ss/%03>Hs %<st %rm %ru %un %Sh/%<A
%mt

logformat squidmime %ts.%03tu %6tr %>a %Ss/%03>Hs %<st %rm %ru %un
%Sh/%<A %mt [%>h] [%<h[

logformat common %>a %ui %un [%tl] "%rm %ru HTTP/%rv" %>Hs %<st
%Ss:%Sh

logformat combined %>a %ui %un [%tl] "%rm %ru HTTP/%rv" %>Hs %<st
}%"Referer}>h" "%{User-Agent}>h" %Ss:%Sh
```

همانطور که می بینید الگوی پیش فرض برای گزارش squid گیری نام دارد.هم اکنون و با توجه به توضیحاتی که در جدول راهنمای کدها بیان شد به راحتی می توانیم پیام های رخداد ایجاد شده را تفسیر و درک نمائیم.خط زیر یکی از این

پیام ها را نشان می دهد:

1284565354.102 147 127.0.0.1 TCP_MISS/200 1786 GET http://www. google.co.in /favicon.ico - FIRST_UP_PARENT/proxy.example.com image/x-icon

اسکوئید پارامترهای بسیاری متنوعی را جهت گزارش گیری و ثبت اطلاعات کاربران فراهم کرده است که میتوان این اطلاعات را با توجه به سیاستهای مورد نظر خود سفارشی کردن نمود.

فرض کنیم می خواهیم زمان ثبت شده همراه با گزارشات ایجاد شده به صورت فرمت استاندارد جهانی آن ثبت شود برای این کار قالب ثبت گزارشات را به صورت زیر تعریف می کنیم:

logformat minimal %tl %>a %Ss/%03>Hs %rm %ru

access_log daemon:/opt/squid/var/logs/access.log minimal

در صورت اضافه کردن تنظیمات بالا ، پیام های رخداد ثبت شده به صورت زیر نمایش داده می شوند:

/۱۱Sep/2010:23:52:33 +0530 127.0.0.1 TCP_MISS/200 GET http//:

en.wikipedia.org/wiki/Main_Page

/۱۱Sep/2010:23:52:34 +0530 127.0.0.1 TCP_MISS/200 GET http//:

en.wikipedia.org/images/wikimedia-button.png

همانطور که مشاهده می کنید زمانهای ثبت شده در این گزارشات به صورت کاملا گویا نشان داده می شوند و تمامی افراد به راحتی می توانند زمان ایجاد پیام ها را مشاهده کنند.

انتخابی کردن گزارشات

گاهی اوقات در سازمانها شرایطی پیش می آید که مایل نیستیم اسکوئید بر فعالیتهای گروهی خاص از کارکنان نظارت داشته باشد و فعالیتهای آنان در وب را دنبال کند مثلا گروهی خاص از مهندسان یک شرکت را تصور کنید که برروی یک پروژه محرمانه مشغول به فعالیت هستند و نباید کوچکترین اطلاعاتی از فعالیتهای آنان در گزارشات ثبت شود.به هرحال این مسئله هر چه باشد اسکوئید برای آن راه حلی ارائه کرده.کلیه ی عملیات ثبت و ذخیره کردن گزارشات کاربران توسط دو دستور access_log و log_access قابل کنترل و مدیریت است تشابه زیاد این دو دستور ممکن است کمی گیج کننده باشد access_log، همواره برای تعیین مسیر ذخیره سازی گزارشهای ایجاد شده و همچنین تعیین فرمتهای مختلف گزارش گیری به کار می رود در حالیکه log_access به منظور نظارت بر محدوده ی گزارش گیری از کاربران و اینکه عمل ثبت رویداد ها

برای کدام دسته از کاربران انجام شود و یا بر گروهی خاص صورت نگیرد.فرض کنیم می خواهیم تمامی فعالیتها و تبادل اطلاعات کاربران با سرورهای شرکت یاهو توسط اسکوئید نادیده گرفته شود و در مقابل میخواهیم تمامی فعالیتهای کاربران با سرورهای شرکت گوگل و فیس بوک در فایلهای جداگانه ای ثبت و ذخیره شوند برای پیاده سازی این سناریو به صوریت زیر عمل می کنیم:

```
acl yahoo dstdomain .yahoo.com
acl google dstdomain .google.com
acl facebook dstdomain .facebook.com
log_access deny yahoo
log_access allow all
access_log /opt/squid/var/logs/google.log squid google
access_log /opt/squid/var/logs/facebook.log squid facebook
access_log /opt/squid/var/logs/access.log
```

درسه خط اول سه acl متفاوت به نامهای yahoo , google , facebook ایجاد شده اند که وظیفه ی آنها شناسایی سرورهای این سه شرکت است در خط چهارم اجازه ی ثبت گزارشات سرورهای یاهو از اسکوئید سلب می شود و در ادامه برای هر یک از سرورهای شرکتهای گوگل و یاهو یک فایل جداگانه ی ثبت گزارش ایجاد می شود که وظیفه ی آنها فقط گزارش گیری از آنهاست و در خط آخر نیز سایر فعالیتهای کاربران با سایر سرورهای موجود در وب ذخیره می شود.همانطور که می بینید این روش برای مدیریت فعالیتهای کاربران در یک سازمان و یا شرکت بسیار مفید است و به راحتی می توان بر فعالتیهای کارکنان نظارت نمود در واقع این روش بررسی و آنالیز گزارشات را بسیار آسان کرده و آنها را در فایلهای جداگانه ای ذخیره می کند تا در مواقع ضروری بتوان به سرعت به آنها دسترسی پیدا کرد.

Referer log

هنگامی که یک کاربر برروی لینک other.example.com برروی وب سایت example.com کلیک می کند در این صورت example.com کاربر را به آدرس other. example.com ارجاع می دهد.زمانی که کاربری از طریق یک آدرس به آدرس دیگری ارجاع داده می شود ، هدر referrer توسط کاربر ارسال می شود.اسکوئید قادر به گزارش گیری از هدر های referrer و استفاده از آنها در الگوهای آنالیز ترافیک شبکه می باشد.

به طور پیش فرض این قابلیت غیر فعال بوده و با استفاده از دستور access_log به همراه ترکیب آن با یک الگوی گزارش

گیری می توان این قابلیت را فعال کرد.برای این کار قالب گزارش گیری ما باید به صورت زیر باشد:

logformat referer %ts.٠٣٪tu %>a %{Referer}>h %ru

این تنظیمات ،یک الگوی جدید گزارش گیری به نام referrer را تعریف می کند که شامل ارائه گزارشاتی چون زمان ایجاد یک درخواست ، آدرس IP کاربر ، نشانی URL ارجاع داده شده و همچنین نشانی درخواست می باشدحال باید الگوی تعریف شده را همراه با دستور access_log به صورت زیر به کار ببریم:

access_log /opt/squid/var/logs/referer.log referrer

پس از اعمال این تنظیمات ، خطوط گزارش گیری ایجاد شده با صورت زیر نمایان می شوند:

۱۲۷.۰.۰.۱ ۱۲۸۴۵۷۶۶۰۱.۸۹۸http://en.wikipedia.org/wiki/Main_Page

http://en.wikiquote.org/wiki/Main_Page

۱۲۷.۰.۰.۱ ۱۲۸۴۵۷۶۶۰۷.۷۳۲http://en.wikiquote.org/wiki/Main_Page

http://upload.wikimedia.org/wikiquote/en/b/bc/Wiki.png

درک اطلاعات موجود در گزارش گیری نوع referrer آسان تر از سایر سبک های گزارش گیری است به طوریکه در ستون اول تاریخ ایجاد گزارش را نشان می دهد که بر اساس الگوی یونیکسی است ، در ستون دوم نشانی IP کاربر را نشان می دهد ،در ستون سوم نشانی لینکی که ارجاع داده می شود را مشخص می کند و بالاخره در ستون چهارم نیز لینکی که ارجاع دهنده با آه ارجاع داده می شود را نشان می دهد.

همانند موارد قبل با استفاده از یک اسکریپت ساده می توان قالب گزارش گیری بالا را ساده تر و قابل فهم تر کرد.اسکریپت مورد استفده در این بخش به صورت زیر است:

```
$cat referer.log | awk '{printf("%s ", strftime("%d/%b/ %Y:%H:%M:%S",$1)); print $2 " " $3 " " $4;}' > referer_human_readable.log
```

پس از اضافه کردن اسکریپت بالا،خطوط ایجاد شده در گزارشات به صورت زیر نمایان می شوند:

/۱۲Sep/ ۱۲۷.۰.۰.۱ ۲۰۱۰:۰۱:۳۶:۰۶http://en.wikipedia.org/wiki/Main_Page

http://en.wikiquote.org/

12/Sep/2010:01:36:12 127.0.0.1 http://en.wikiquote.org/wiki/Main_Page

http://upload.wikimedia.org/wikiquote/en/b/bc/Wiki.png

User agent log

تمامی درخواستهای دریافت شده از سوی کاربران حاوی هدر user-agent است که یکی از هدر های http و شامل اطلاعاتی در مورد کاربر مورد نظر است که یکی از این اطلاعات نوع مرورگر مورداستفاده است.اسکوئید قادر است ، این هدر ها را در قالب گزارش در دیسک سخت نگهداری کند دستوری که برای این منظور تعیین شده است useragent_log نام دارد.نحوه به کارگیری این دستور مشابه cache_log و access_log است و به صورت زیر به کار می رود:

useragent_log /opt/squid/var/logs/useragent.log

با اضافه کردن خط بالا به فایل پیکربندی می توان تمامی این هدر ها که شامل اطلاعاتی در مورد نوع و نسخه ی مرورگر مورداستفاده است را در یک فایل گزارش به نام useragent.log ذخیره نمود.نمونه ی این گزارشها به صورت زیر است:

127.0.0.1 [12/Sep/2010:01:55:33 +0530] "Mozilla/5.0 (X11; U; Linux

i686; en-US; rv:1.9.2.6) Gecko/20100625 Firefox/3.6.6 GTB7.1"

127.0.0.1 [12/Sep/2010:01:55:33 +0530] "Mozilla/5.0 (X11; U; Linux

i686; en-US; rv:1.9.2.6) Gecko/20100625 Firefox/3.6.6 GTB7.1

GoogleToolbarFF 7.1.20100830 GTBA"

با دقت در خطوط بالا می توان دریافت که هریک از کاربران از چه مرورگری به منظور بازدید وب سایتها استفاده می کنند.مزیت اصلی این نوع گزارش گیری ،به دست آوردن اطلاعاتی در مورد محبوبترین مرورگر مورد استفاده ی کاربران است که طی یک زمان مشخص می توان به آن پی برد.

خلاصه کردن فرایندگزارش گیری

سرویس دهنده اسکوئید در طی فرایند گزارش گیری از کاربران جزئیات زیادی را در گزارش خود اضافه می کند که بسیاری از آنها خارج از حد معمول است و درک آن برای افرادی که تاکنون با اسکوئید آشنایی نداشته اند ممکن است مشکل باشد.دستور common برای این موضوع ایجاد شده است و وظیفه ی آن تبدیل گزارش های معمولی اسکوئید به الگوهای رایج گزارش گیری توسط سرویس دهنده های پرطرفدار لینوکس مانند Apache است.در واقع این دستور جزئیاتی را که به طور معمول همراه با گزارشات توسط اسکوئید تهیه می شوند را حذف می کند.بدین منظور خط زیر را به فایل پیکربندی اضافه کنید:

access_log daemon:/opt/squid/var/logs/access.log common

این دستور گزارشهای access_log را مانند گزارش گیری های وب سرورهای معروفی مانند آپاچی تهیه می نماید که نمونه ی آن به صورت زیر است:

127.0.0.1 - - [13/Sep/2010:17:38:57 +0530] "GET http://www.google.com/

HTTP/1.1" 200 6637 TCP_MISS:FIRSTUP_PARENT

127.0.0.1 - - [13/Sep/2010:17:40:11 +0530] "GET http://example.com/

HTTP/1.1" 200 1147 TCP_HIT:HIER_NONE

127.0.0.1 - - [13/Sep/2010:17:40:12 +0530] "GET http://example.com/

favicon.ico HTTP/1.1" 404 717 TCP_MISS:FIRSTUP_PARENT

همانطوریکه واضح است دستور common جزئیاتی راکه همراه با گزارشهای معمول اسکوئید وجود دارد را حذف کرده و آن را به صورت خلاصه تر نشان می دهد.

گرد کردن فایلهای ثبت رویداد

با گذشت زمان،حجم فایلهای ثبت رویداد اسکوئید بسیار زیاد می شوند و فضای زیادی از دیسک سخت را اشغال می کنند. برای حل این مشکل باید فایلهای حجیم شده را از طریق گرد کردن کم حجم کرده و گزارش های تهیه شده را در طول مدت زمان مشخص مانند یک یا دو هفته یک بار که دیگر ارزش چندانی ندارند حذف کرد.دستور rotate به صورت زیر به کار می رود:

Squid –k rotate

با اجرای این دستور log های موجود را گرد می کند و با تغییر نام ، آنها در همان پوشه ی نگهداری log ها ذخیره می کند.تعداد فایلهای گرد شده ی ایجاد شده توسط دستور logfile_rotate در فایل پیکربندی تعیین می شود که مقدار پیش فرض آن 10 است یعنی تعداد 10 فایل گرد شده را به طور همزمان نگهداری می کند.

Cache Store Log

یکی دیگر از سیستم های گزارش گیری اسکوئید store log نام دارد.هنگامی که قابلیت کش کردن اطلاعات فعال است ، با فعال کردن store log تمامی اطلاعات مربوط به object های کش شده در این فایل نگهداری می شود و مشابه

access.log عمل می کند این نوع گزارش گیری کاملا اختیاری بوده و از اهمیت پایین تری نسبت به دیگر موارد برخوردار است و به طور پیش فرض نیز این گزینه غیر فعال است به هر حال درصورت تمایل می توانید با استفاده از دستور cache_store_log مشابه موراد قبلی از این سیستم گزارش گیری استفاده نمایید.

خلاصه فصل

این فصل به طور اختصاصی انواع فایلهای ثبت رویداد اسکوئید را به همراه ویژگیها و کاربردهای هر یک به طور مفصل شرح داده شد.همچنین در مورد فرمتهای مختلف log های اسکوئید ،چگونگی راه اندازی و پیکربندی آنها مطالب مفصلی ارائه شد.cache_log های به طور اختصاصی تمامی فعالیتهای اسکوئید را زیر نظر دارد و هرگونه اشکال و یا خطایی در عملکرد اجرایی اسکوئید را به طور مفصل گزارش می دهد در مقابل ،access_log ها اختصاصا برای ثبت فعالیتهای کاربران ایجاد شده و تمامی فعالیتهای کاربران در شبکه را گزارش می دهد.useragent_log ها یکی از log های اسکوئید بوده که به منظور بررسی هدر user-agent مورد استفاده قرار می گیرد.همچنین در این فصل در مورد چگونگی گرد کردن فایلهای ثبت رویداد و حذف دوره ای آنها را به منظور جلوگیری از هدر رفتن فضای دیسک سخت مطالبی مطرح شد.

به طور کلی مطالب زیر در این فصل بررسی شد:

- نصب و پیکربندی وب سرور آپاچی به منظور دسترسی و استفاده از رابط گرافیکی cachemge.cgi
- بدست آوردن اطلاعات جامعی در مورد وضعیت پراکسی سرور در حال اجرا
- آشنایی با ابزارهای آنالیز فایلهای ثبت رویداد
- نصب و راه اندازی ابزار calamaris و sarg به منظور بررسی و آنالیز فایلهای ثبت رویداد

فصل ۶

مدیریت اسکوئید و ترافیک شبکه

در فصل گذشته در مورد فایلهای ثبت رویداد و انواع آنها مطالبی را آموختیم همانطور که می دانیم آنالیز و بررسی این فایلها به دلیل حجم زیاد محتوای موجود در آنها کاری دشوار و بعضا غیر ممکن است این در حالی است که این گزارشات خود حاوی اطلاعات ارزشمندی در مورد چگونگی عملکرد اسکوئید و همچنین علاقه مندی های کاربران است.در این فصل با رابط گرافیکی مشاهده وضعیت اسکوئید و ابزارهای مختلف آنالیز گزارشات ایجاد شده توسط اسکوئید آشنا خواهیم شد.

مطالب مورد بحث در این فصل شامل موارد زیر است:

- چگونگی نصب و استفاده از رابط گرافیکی Cache Manager
- نصب و راه اندازی ابزارهای مختلف آنالیز فایلهای ثبت رویداد

Cache manager

همانطور که در فصلهای پیش گفته شد cache manager رابط گرافیکی مدیریت اسکوئید است.این رابط گرافیکی به طور پیش فرض به همراه اسکوئید ارائه می شود بدین معنی که با وجود cache manager هیچ گونه نرم افزار جانبی ای مورد نیاز نیست.cache manager در حقیقت یک محیط گرافیکی تحت وب به منظور بررسی وضعیت اسکوئید از جمله میزان منابع سخت افزاری ،میزان بازده کش سرور و سایر موارد تخصصی به کار می رود که از طریق آن می توان به اطلاعات بسیار وسیع و مفیدی در مورد عملکرد پراکسی سرور دست پیدا کرد.پیش از آنکه بتوان از این رابط استفاده کرد باید از راه اندازی و پیکربندی وب سرور به همراه پراکسی سرور اطمینان حاصل نمود.

نصب وب سرور Apache

هر چند راه اندازی یک وب سرور خارج از حیطه ی کاری این کتاب است اما به دلیل اینکه یکی از پیش نیازهای راه اندازی cache managerاست به چگونگی نصب و راه اندازی آن می پردازیم.وب سرور آپاچی بدون اغراق قدرتمندترین و پرطرفدارترین وب سرور مورداستفاده در حال حاظر است و به طور کاملا رایگان توسط تمامی توزیعهای لینوکس ارائه می شود. برای نصب آپاچی برروی توزیعهای ردهت و سایر توزیعهای پایه ای آن از دستورyum install httpd استفاده کنید.برای نصب آپاچی برروی سرورهای برپایه سیستم عامل دبیان و توزیعهای پایه ای آن مانند اوبونتو از دستور apt-get install apache2 استفاده کنید.برای نصب آپاچی برروی سایر توزیعهای لینوکس با مراجعه به مستندات رسمی آنها با نحوه نصب و راه اندازی آن آشنا شوید.

پیکربندی وب سرور آپاچی جهت مشاهده رابط گرافیکی Cavhe Manager

پس از نصب وب سرور باید آن را جهت استفاده از cachemgr.cgi پیکربندی کنیم.فایل اصلی cachemgr معمولا در مسیر $\{prefix\}/libexec/cachemgr.cgi وجود دارد که منظور از {prefix} دایرکتوری ای است که در توزیعهای مختلف ممکن است متفاوت باشد.برای اینکه آپاچی بتواند به cachemgr.cgi دسترسی داشته باشد باید خطوط زیر را در یک فایل متنی به نام squid-cachemgr.conf ذخیره کرده و آن را به دایرکتوری پیکربندی آپاچی که معمولا در مسیر etchttpd/conf.d/ و یا etc/apache2/conf.d/ منتقل کنیم :

```
ScriptAlias /Squid/cgi-bin/cachemgr.cgi /opt/squid/libexec/cache. cgi

# Only allow access from localhost by default
```

```
<Location /Squid/cgi-bin/cachemgr.cgi>
 order allow,deny
 allow from localhost
 # If we want to allow access to cache manager from 192.0.2.25,
 # uncomment the following line
 # allow from 192.0.2.25
 # Add additional allowed hosts as needed
 # allow from .example.com
</Location>
```

پس از آنکه این فایل در دایرکتوری اصلی پیکربندی آپاچی قرار گرفت ،به منظور اعمال تغییرات باید وب سرور را دوباره راه اندازی کنیم برای این کار از دستور apachectl graceful و یا service apache2 restart استفاده می کنیم.به منظور کسب اطلاعات بیشتر در مورد چگونگی راه اندازی و پیکربندی وب سرور آپاچی به http://httpd.apache.org/docs/current/configuring.html مراجعه نمائید.

دسترسی به رابط کاربری cache manager

پیش از آنکه بتوانیم از رابط cache manager استفاده کنیم باید اسکوئید را جهت دسترسی به آن پیکربندی کنیم زیرا در حالت پیش فرض اسکوئید اجازه ی دسترسی به cache manager را نخواهد دادبدین منظور دو دستور به نامهای cache_mgr و cachemgr_passwd پیش بینی شده است.cache_mgr در واقع برای تعیین نشانی ایمیل مدیر شبکه که بعدا به عنوان نام کاربری هم از آن استفاده می شود به کار می رود همچنین در مواقعی که اسکوئید از کار بیفتد به طور خودکار یک ایمیل به این نشانی ارسال می شود و مدیر شبکه را از این موضوع آگاه می کند.مقدار پیش فرض آن webmaster است که برای تغییر آن به شیوه زیر عمل می کنیم:

```
cache_mgr admin@example.com
```

این خط نشانی ایمیل مدیر شبکه را به آدرس مورد نظر شما تغییر می دهد فراموش نکنید این همان ایمیلی است که در صفحات خطای اسکوئید نیز نمایان می شود. دستور cachemgr_passwd به منظور کنترل دسترسی به رابط گرافیکی به کار می رود توسط این دستور می توان علاوه بر تعیین رمز ورود ، دسترسی به قسمتهای مختلف cache manager را محدود کرد. ساختار کلی این دستور به صورت زیر است:

cachemgr_passwd PASSWORD ACTION ACTION ...

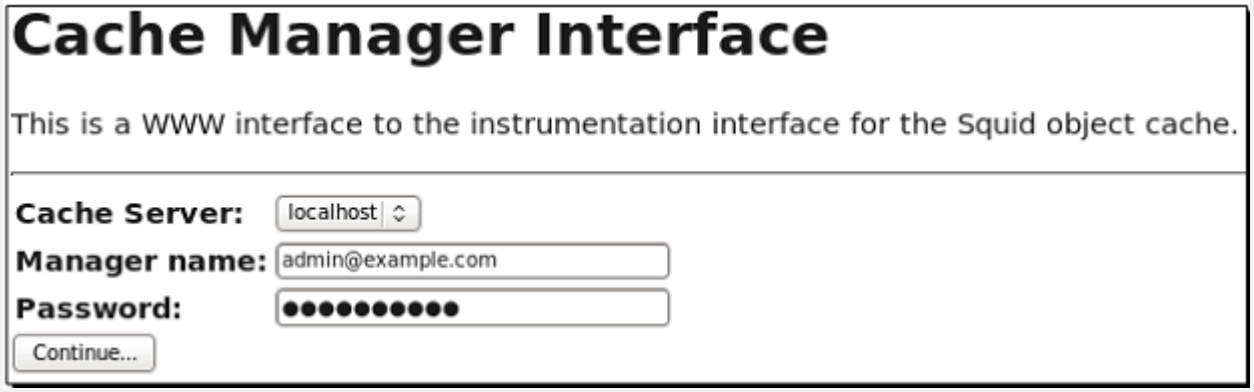
به جای پارامتر PASSWORD رمز ورود مورد نظر قرار می گیرد همچنین به جای پارامتر ACTION می توان سطح دسترسی را تعیین کرد در صورتیکه به جای آن عبارت all قرار گیرد ،کاربر مورد نظر با وارد کردن نام کاربری و رمز ورود می تواند به تمامی قسمتهای cache manager دسترسی داشته باشد به طور معمول تنظیمات این دستور به صورت زیر است:

cachemgr_passwd s3cr3tP4sS all

با قرار دادن ابن خط در فایل پیکربندی کاربران با وارد کردن این رمز عبور و نام کاربری تعیین شده می توانند به تمامی قسمتهای رابط کاربری دسترسی داشته باشند.

ورود به cache manager

برای ورود به رابط گرافیکی cache manager آدرس http://localhost/Squid/cgi-bin/cachemgr.cgi را در یک مرورگر وارد کنید در این آدرس به جای عبارت localhost آدرس ip پراکسی سرور مورد نظر را وارد کنید.در صورتکیه تمامی تنظیمات شما به خوبی اعمال شده باشند پس از وارد کردن آدرس بالا صفحه ای گرافیکی مشابه زیر نمایان می شود:



تصویر ۶-۱ صفحه ی اول ورود به رابط گرافیکی Cache Manager

در قسمت Manager name: آدرس ایمیلی که قبلا تعیین شده بود یعنی admin@example.com را وارد می کنیم و در قسمت Password نیز رمز عبور تعیین شده در قسمت cachemgr_password را وارد می کنیم و برروی دگمه ی Continue در قسمت پایین سمت چپ کلیک کرده و صفحه ی اصلی رابط گرافیکی cache manager برای شما نمایان می شود:

Cache Manager menu for localhost:

- This Cachemanager Menu
- Shut Down the Squid Process (hidden).
- Reconfigure the Squid Process (hidden).
- Toggle offline_mode setting (hidden).
- Current Squid Configuration (hidden).
- AS Number Database
- CARP information
- peer userhash information
- peer sourcehash information
- Callback Data Registry Contents
- comm_incoming() stats
- Cache Client List
- Delay Pool Levels
- Async IO Function Counters
- DISKD Stats

تصویر ۶-۲ صفحه ی اصلی رابط گرافیکی Cache Maagerرا نشان می دهد

تصویر بالا تنها قسمت کوچکی از گزینه های موجود در صفحه ی اصلی را نشان می دهد و تعداد گزینه های موجود در آن به نسخه ی مورداستفاده ی اسکوئید و تعداد ویژگیهای فعال شده هنگام کامپایل آن بستگی دارد. در ادامه به بررسی برخی از مهمترین گزینه های موجود در آن می پردازیم:

General Runtime Information

با کلیک برروی این گزینه صفحه ی دیگری نمایان می شود که حاوی اطلاعاتی در مورد وضعیت اسکوئید در حال اجر است .

Start Time:	Mon, 20 Sep 2010 03:38:36 GMT
Current Time:	Tue, 25 Sep 2010 03:05:23 GMT

```
Connection information for squid:
        Number of clients accessing cache:      1146
        Number of HTTP requests received:       60396670
        Number of ICP messages received:        0
        Number of ICP messages sent:    0
        Number of queued ICP replies:   0
        Request failure ratio:   0.08
        Average HTTP requests per minute since start:   5257.9
        Average ICP messages per minute since start:    0.0
        Select loop called: 1978046739 times, 0.348 ms avg
Cache information for squid:
        Request Hit Ratios:     5min: 34.5%, 60min: 33.9%
        Byte Hit Ratios:        5min: 3.8%, 60min: 14.2%
        Request Memory Hit Ratios:      5min: 20.8%, 60min: 21.4%
        Request Disk Hit Ratios:        5min: 28.9%, 60min: 29.2%
        Storage Swap size:      357549416 KB
        Storage Mem size:       2097160 KB
        Mean Object Size:       55.60 KB
        Requests given to unlinkd:      0
```

تصویر ۶-۳ جدول حاوی اطلاعاتی در مورد وضعیت اسکوئید در حال اجرا را نشان می دهد.

در جدول اول در ابتدای صفحه زمان اجرای اسکوئید و زمان فعلی را نشان می‌دهد که درواقع نشان دهنده مدت زمانی است که اسکوئید بدون راه اندازی مجدد مشغول فعالیت بوده است. در سطر بعدی اطلاعات جامعی در مورد تعداد درخواستهای پاسخ داده شده و همچنین تعداد کاربرانی که درخواستهای خود را از اسکوئید دریافت کرده‌اند نمایش داده می شود و اطلاعاتی در مورد تعداد اتصالات از طریق پروتکل HTTP و ICP و همچنین میانگین درخواستهای دریافتی نمایش داده می شود.

در سطر بعدی یعنی Cache information for squid، اطلاعات بسیاری مفیدی در مورد عملکرداسکوئید از نظر میزان HIT Rate آن در دیسک سخت و حافظه ی رم و همچنین میانگین صرفه جویی اسکوئید و میزان حافظه ی باقی‌مانده از دیسک سخت که برای مخزن کش در نظر گرفته شده را نمایش می‌دهد.در سطرهای بعدی اطلاعات جامعی در مورد میزان حافظه ی مصرف شده ،میزان مصرف CPU و تعداد اشیاء موجود برروی دیسک سخت و حافظه ی رم ،تعداد اشیاء پربازدید همراه با جزئیات نمایش داده می‌شوند.این گزینه حاوی اطلاعات جامع و مفیدی در مورد عملکرد کلی اسکوئید است و منبع قابل اطمینانی برای مشاهده وضعیت کش سرور می باشد.

IP Cache Stats and Contents

با کلیک برروی این گزینه صفحه‌ای دیگر باز می شود که شامل اطلاعاتی در مورد تعداد ip های ورودی و کش شده توسط اسکوئید است این صفحه درواقع بازتاب تنظیمات شما در قسمتهای ipcache_size و ipcache_high و ipcache_low است که قبلاً در فایل پیکربندی صورت گرفته است.اطلاعاتی که در این صفحه نشان داده می شود مشابه موارد زیر است:

IP Cache Statistics:

IPcache Entries: 14550

IPcache Requests: 139729579

IPcache Hits: 119273350

IPcache Negative Hits: 2619823

IPcache Numeric Hits: 8339299

IPcache Misses: 9496827

IPcache Invalid Requests: 280

نام گزینه ی مورد نظر	توضیحات
تعداد ip های کش شده ،این تعداد توسط دستور ipcache_size قابل کنترل است.	IPcache Entries
تعداد کل درخواستهای ارسالی به اسکوئید به منظور ترجمه آدرس دامنه آنها به نشانی IPتعیین شده برای آنها .	IPcache Requests
تعداد درخواستهایی که اسکوئید توانسته است از طریق کش کردن نامهای دامنه به آنها پاسخ دهد ودر واقع بدون مراجعه به DNS سرورمرجع به آنها پاسخ دهد.	IPcache Hits
تعداد DNS های کش شده که به هر صورتی دچار اشکال شده اند و نیاز به بازخوانی دارند.	IPcache Negative Hits
زمانی که کاربری به جای نام سایت از آدرس ip معادل آن استفاده کند.	IPcache Numeric Hits
درخواستهایی که نام DNS ی که برای ترجمه ی نشانی وب سایت کار می رود در کش موجود نباشد و اسکوئید مجبور است نام dns مورد نظر را به طور مستقیم از منبع اصلی آن دریافت کند.	IPcache Misses
در صورتیکه کاربر آدرس یک وب سایت را اشتباه وارد نماید باعث بروز این خطا می شود.	IPcache Invalid Requests

علاوه بر گزارشات فوق اسکوئید می تواند به طور همزمان آدرس سایت هایی را که کاربران هم اکنون در خواست کرده اند را به را در این صفحه به همراه آدرس ip آنها به صورت زیر نمایش دهد:

IP Cache Contents:

Hostname Flg lstref TTL N(b)

69.171.229.14-OK

us.mg6.mail.yahoo.com 2 233 4(0) 66.196.66.157-OK

66.196.66.212-OK

66.196.66.213-OK

66.196.66.156-OK

fa.wikipedia.org 2 137 1(0) 208.80.154.225-OK

en-maktoob.yahoo.com 2 42 2(0) 87.248.122.122-OK

در این جدول اطلاعاتی شامل نام دامنه و میزبان مورد نظر ، تعداد IPهای کش شده برای یک آدرس دامنه ، مدت زمان پاسخگویی ، تعداد درخواستهای موجود برای یک دامنه نمایش داده می شود.

HTTP Header Statistics

در فصل های گذشته مطالبی در مورد HTTP Header ها آموختیم این گزینه تمامی آمارهای دریافت انواع مختلف HTTP Header ها را نگهداری می کند با کلیک برروی HTTP Header Statistics در صفحه ی اصلی cache manager صفحه ی دیگر گشوده خواهد شد که حاوی اطلاعات کاملی در مورد میزان استفاده از هدر های HTTP است در شکل زیر یکی از جدول های این صفحه را مشاهده می کنید:

Header Stats: request

Field type distribution

id	name	count	#/header
0	Accept	80959818	0.88
1	Accept-Charset	74816256	0.81
2	Accept-Encoding	77445395	0.84
3	Accept-Language	76570620	0.83
4	Accept-Ranges	3250	0.00

تصویر ۶-۴ اطلاعاتی درمورد جزئیات هدر های HTTP را نمایش می دهد.

به طور کلی این جدول و سایر جدولهای این صفحه اطلاعاتی به شرح زیر را در اختیار ما قرار می دهند:

- ستون اول شامل id است که برای شناسایی انواع هدر ها توسط اسکوئید به کار می رود.
- ستون دوم نام هدر های HTTP را نشان می دهد.
- ستون سوم تعداد هدر هایی را که تمامی کاربران دریافت کرده اند را نشان می دهد در واقع این ستون تعداد کل هدر هایی از هر نوع را که توسط کاربران دریافت شده است را نمایش می دهد.
- ستون چهارم آمار ستون سوم را به صورت درصد بیان می کند.

Traffic and Resource Counters

این لینک میزان داده های مبادله شده میان کاربران و کش سرور را به صورت مجزا در ردیف های جدا از هم نشان می دهد.اطلاعات موجوددر این گزینه تخصصی تر هستند .تصویر زیر نمونه ای از اطلاعات موجود در این گزینه را نشان می دهد:

```
client_http.requests = 61265446
client_http.hits = 22386052
client_http.errors = 1334219
client_http.kbytes_in = 196353418
client_http.kbytes_out = 1726843361
client_http.hit_kbytes_out = 292757460
server.all.requests = 40989383
server.all.errors = 0
server.all.kbytes_in = 1417894692
server.all.kbytes_out = 184760475
server.http.requests = 35932317
server.http.errors = 0
server.http.kbytes_in = 1171528089
server.http.kbytes_out = 59247182
server.ftp.requests = 1249
server.ftp.errors = 0
server.ftp.kbytes_in = 4336291
server.ftp.kbytes_out = 228
```

تصویر ۶-۵ اطلاعات جامعی در مورد میزان ترافیک مبادله شده و منابع مورد استفاده نشان می دهد.

اطلاعات قابل مشاهده در اینجا تعداد کل درخواستهای HTTP دریافتی از کاربران ،میزان HIT Rate اسکوئید که در اینجا این مقدار 22386052 است که بسیار قابل توجه است همچنین تعداد خطاهای ایجاد شده ،حجم داده های دریافتی و ارسالی را نشان می دهد که با مطالعه ی این آمار و ارقام می توان از میزان عملکرد و موفقیت کش سرور مطلع شد.با مشاهده آمار ارائه شده در مقابل هر گزینه می توان از عملکرد کلی پراکسی سرور اسکوئید در تبادل داده از طریق پروتکل های مختلف ، میزان

بروز خطا در تبادل اطلاعات میان سرور و کاربر ، میزان دقیق ترافیک ورودی و خروجی از کش سرور و همچنین میزان کارایی مطلع شد.

Cache Client List

اسکوئید در هر شبانه روز آمار کاربرانی را که با کش سرور تبادل داده داشته اند را نگهداری می کند .فراموش نکنید این اطلاعات مربوط به ۲۴ ساعت گذشته است.با کلیک برروی گزینه cache client list در منوی اصلی صفحه ای مانند صفحه زیر را مشاهده می کنید:

```
Cache Clients:
Address: 10.1.98.78
Name: 10.1.98.78
Currently established connections: 0
    ICP Requests 0
    HTTP Requests 2255
        TCP_HIT                 651  29%
        TCP_MISS               1131  50%
        TCP_REFRESH_HIT          56   2%
        TCP_REFRESH_MISS          3   0%
        TCP_CLIENT_REFRESH_M      6   0%
        TCP_IMS_HIT              26   1%
        TCP_MEM_HIT             382  17%

Address: 10.1.40.190
Name: 10.1.40.190
Currently established connections: 0
    ICP Requests 0
    HTTP Requests 1510
        TCP_HIT                 227  15%
        TCP_MISS                856  57%
        TCP_REFRESH_HIT         111   7%
        TCP_REFRESH_MISS         14   1%
        TCP_CLIENT_REFRESH_M     21   1%
        TCP_IMS_HIT             183  12%
        TCP_NEGATIVE_HIT         13   1%
        TCP_MEM_HIT              84   6%
        TCP_DENIED                1   0%
```

تصویر ۶-۶ اطلاعاتی را در مورد کاربرانی که طی ۲۴ ساعت گذشته با کش سرور تبادل داده داشته اند را نمایش می دهد.

این صفحه علاوه بر نام و آدرس کاربران ، اطلاعات مفیدی درمورد میزان درخواستهایی که کاربر به سوی کش سرور ارسال کرده و همچنین میزان hit rate ووضعیت فعال یا غیر فعال بودن کاربر و همچنین تعداد اتصالات برقرار شده میان کاربر و سرور را نشان می دهد.

Memory Utilization

در این بخش اطلاعات مفید و جالی در مورد میزان صرف حافظه رم توسط بخشهای مختلف موجود در فایل پیکربندی نمایش داده می شود:

Pool	Obj Size	Allocated			
	(bytes)	(#)	(KB)	high (KB)	high (hrs)
acl	64	35	3	3	199.06
acl_deny_info_list	32	1	1	1	199.06
acl_ip_data	24	98	3	3	170.29
acl_list	24	46	2	2	199.06
acl_name_list	40	1	1	1	199.06
acl_time_data	24	3	1	1	199.06

تصویر ۶-۷ میزان حافظه مصرف شده توسط بخش های مختلف را نشان می د هد.

اطلاعات ارائه شده در این بخش به توسعه دهندگان ابزارهای وب کمک می کند تا از بخشهای پرمصرف حافظه توسط پراکسی سرور آگاه شوند همانطور که می بینید اطلاعات جامعی در مورد میزان حافظه مصرق شده توسط ابزارهای مهمی چون aclهای مختلف نمایش داده می شود.

نکته: آمار ارئه شده در مورد میزان مصرف حافظه در این جدول با آمار کلی مصرف حافظه رم در اسکوئید متفاوت بوده و میزان کلی آن بیشتر از این ارقام است زیرا در این جدول میزان مصرف حافظه توسط سایر بخش های اسکوئید را نمایش نخواهد داد.

Internal DNS Statistics

همانطوری که قبلا هم گفته شد اسکوئید به منظور سرعت بخشیدن به فرایند تبدیل آدرس های دومین به ip از یک dns client داخلی استفاده می گیرد که وظیفه آن نگهداری و یا کش کردن آدرس های دومین به همراه resolver آنها و استفاده از آنها در موارد مشابه است تا میزان مراجعه به سرورهای خارجی کاهش یابداین گزینه اطلاعات مفیدی در مورد وضعیت DNS server های مورد استفاده توسط اسکوئید را نشان می دهد.

```
Internal DNS Statistics:

The Queue:
                                DELAY SINCE
  ID    SIZE SENDS FIRST SEND LAST SEND
------ ---- ----- ---------- ---------
0xaf24   34     1      0.157     0.157
0xa717   42     1      0.542     0.542

Nameservers:
IP ADDRESS        # QUERIES # REPLIES
--------------- --------- ---------
192.168.36.204    1255839   1254437
192.168.36.222      21820     21682

Rcode Matrix:
RCODE ATTEMPT1 ATTEMPT2 ATTEMPT3
    0  3011754     2209      283
    1        0        0        0
    2    75075    72051    71697
    3  1036829      810       66
    4        0        0        0
    5        0        0        0
```

تصویر ۶-۸ اطلاعات مهمی در مورد وضعیت DNSسرور مورد استفاده توسط اسکوئید را نشان می دهد.

در صورتکیه قبلا در فایل پیکربندی از گزینه dns_nameservers استفاده کرده اید این صفحه اطلاعات جامعی در مورد وضعیت DNS های مورد استفاده در اختیار شما قرار می دهد این اطلاعات شامل تعداد کل درخواستهای ارسال شده به آنها و تعداد پاسخ های موفقیت آمیز توسط هر یک را نشان می دهد در واقع اگر بخواهیم از وضعیت هرکدام از DNS Server های مورد استفاده توسط اسکوئید آگاه شویم این گزینه بهترین انتخاب خواهد بود.

Store Directory Stats

این گزینه تمامی اطلاعات مورد نیاز در مورد میزان حافظه مخزن کش مصرف شده توسط اسکوئید و همچنین فضای باقیمانده را در اختیار شما قرار می دهد با کلیک برروی این گزینه اطلاعاتی مشابه خطوط زیر نمایان می شود:

Store Directory Statistics:

Store Entries : 2951424

Maximum Swap Size : 204800000 KB

Current Store Swap Size: 59489376 KB

Current Capacity : 29% used, 71% free

Store Directory #0 (aufs): /var/spool/squid3/cache

FS Block Size 4096 Bytes

First level subdirectories: 16

Second level subdirectories: 256

Maximum Size: 51200000 KB

Current Size: 45381844 KB

Percent Used: 88.64%

Filemap bits in use: 2274738 of 4194304 (54%)

Filesystem Space in use: 67952468/303274376 KB (22%)

Filesystem Inodes in use: 3663541/19259392 (19%)

Flags:

Removal policy: lru

LRU reference age: 170.48 days

Store Directory #1 (aufs): /var/spool/squid3/cache01

FS Block Size 4096 Bytes

First level subdirectories: 16

Second level subdirectories: 256

Maximum Size: 153600000 KB

Current Size: 14107532 KB

Percent Used: 9.18%

Filemap bits in use: 676588 of 1048576 (65%)

Filesystem Space in use: 67952468/303274376 KB (22%)

Filesystem Inodes in use: 3663541/19259392 (19%)

Flags: SELECTED

Removal policy: lru

LRU reference age: 33.07 days

در قسمت اول از گرازشات وضعیت کلی مخزن کش سرور را از نظر میزان فضای مصرف شده و فضای باقیمانده را به صورت در صد و کیلو بایت نشان می دهد در بخش های بعدی هر یک از دایرکتوریهای ایجاد شده به وسیله دستور squid –z را به همراه جزئیات شامل میزان ظرفیت ،مسیر ایجاد،نوع Removal Policy انتخاب شده ،همچنین مدت زمان ایجاد آن برحسب روز را نشان می دهد به طورکلی این گزینه یکی از مهمترین گزینه هایی است که در cache manager وجود دارد و مدیر شبکه باید به طور مداوم آن را بررسی کرده تا از میزان فضای باقیمانده مخزن کش اطلاعات کاملی را بدست آورد.

Current Squid Configuration

با انتخاب این گزینه تنظیمات فعلی پیکربندی اسکوئید نمایش داده می شود شاید در نگاه اول تعداد زیاد خطوط عجیب به نظر

برسد اما به یاد داشته باشید برخی از تنظیمات پیش فرضی که شما در آنها غییری ایجاد نکرده اید به صورت پیش فرض به وسیله ویرایشگرخای متنی قابل نمایش نیستند اما در این گزینه تمامی تنظیمات پیش فرض و تنظیمات اعمال شده توسط کاربر به صورت کامل نمایان خواهد شد نمونه فایل پیکربندی به صورت زیر است:

```
authenticate_cache_garbage_interval 3600 seconds
authenticate_ttl 3600 seconds
authenticate_ip_ttl 0 seconds
acl all src ::/0
acl QUERY urlpath_regex cgi-bin \?

acl Safe_ports port 80 20 21 443 70 210 8080 280 488 591 777 901 1024-65535
acl purge method PURGE
acl CONNECT method CONNECT
follow_x_forwarded_for Deny all
acl_uses_indirect_client on
delay_pool_uses_indirect_client on
log_uses_indirect_client on
http_access Deny google-image1
http_access Allow manager localhost
http_access Allow manager local
http_access Deny manager
http_access Allow purge localhost
http_access Allow purge local
http_access Deny purge
http_access Deny !Safe_ports
http_access Deny CONNECT !SSL_ports
http_access Allow localhost
http_access Allow local
http_access Deny all
icp_access Deny all
htcp_access Deny all
htcp_clr_access Deny all
miss_access Allow all
ident_lookup_access Deny all
http_port 0.0.0.0:3128 intercept name=3128 connection-auth=on
http_port [::]:8080 name=8080 connection-auth=on
qos_flows disable-preserve-misdead_peer_timeout 10 seconds
forward_max_tries 10
hierarchy_stoplist cgi-bin
hierarchy_stoplist ?
cache_mem -1149239296 bytes
maximum_object_size_in_memory 524288 bytes
memory_replacement_policy lru
cache_replacement_policy lru
cache_dir aufs /var/spool/squid3/cache 50000 16 256 IOEngine=DiskThreads
cache_dir aufs /var/spool/squid3/cache01 150000 16 256 IOEngine=DiskThreads
store_dir_select_algorithm least-load
max_open_disk_fds 0
minimum_object_size 0 bytes
```

```
maximum_object_size 20971520 bytes
cache_swap_low 95
cache_swap_high 98
access_log /var/log/squid/access.log
logfile_rotate 2
emulate_httpd_log off
log_ip_on_direct on
mime_table /usr/local/squid/etc/mime.conf
log_mime_hdrs off
pid_filename /usr/local/squid/var/run/squid.pid
log_fqdn off
client_netmask ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
strip_query_terms off
buffered_logs off
cache_log /var/log/squid/cache.log
debug_options ALL,1
coredump_dir var/spool/squid
ftp_user Squid@
ftp_list_width 32
ftp_passive on
ftp_epsv_all off
ftp_epsv on
ftp_eprt on
ftp_sanitycheck on
ftp_telnet_protocol on
diskd_program /usr/local/squid/libexec/diskd
unlinkd_program /usr/local/squid/libexec/unlinkd
url_rewrite_children 5
url_rewrite_concurrency 0
url_rewrite_host_header on
url_rewrite_bypass off
cache Deny QUERY
```

همانطور که می بینید در حلت عادی بسیاری از این دستورات پنهان شده اند و ما آنها را در فایل پیکربندی نمی بینیم!اما در این صفحه به طور کامل قابل مشاهده هستند. رابط گرافیکی Cache manager دارای گزینه های بسیار متنوعی هستند که برخی از مهمترین آنها در اینجا معرفی شدند البته هرکدام از گزینه های موجود در آن دارای اطلاعات مفیدی در مورد وضعیت پراکسی سرور و عملکرد آن است البته بعضی از گزینه های آن به شرطی مفید هستند که آن قابلیت قبلا در اسکوئید فعال شده باشد مثلا گزینه peer cache statistics تنها زمانی مفید است که کش سرور شما با سایر کش سرورهای همسایه در ارتباط باشد به هر حال این رابط گرافیکی به عنوان ابزاری استاندارد که توسط توسعه دهندگان اسکوید به طور رسمی عرضه می شود می تواند راه گشای مدیران شبکه در امر مدیریت کش سرور باشد.

ابزارهای آنالیز LOG های اسکوئید

به منظور دسترسی آسان و سریع به فایلهای ثبت رویداد و همچنین کسب اطلاعات مورد نظر بهتر است از یک نرم‌افزار مدیریت

و آنالایز LOG های اسکوئید استفاده کنیم این نرم‌افزارها تمامی اطلاعات موجود در access.log را پس از مرتب سازی و آنالیز نتایج از طریق یک رابط گرافیکی ساده در اختیار کاربر قرار می‌دهند.ابزارهای گوناگونی بدین منظور ایجاد شده‌اند که در ادامه برخی از آن‌ها معرفی خواهند شد.

Squid Analysis Report Generator (sarg)

یکی از ابزارهای کوچک و کاربردی در این زمینه Sarg نام دارد.این ابزار کوچک و رایگان همواره از طریق وب گاه رسمی آن یعنی http://sarg.sourceforge.net/ قابل دریافت است همچنین بسته های آماده نصب آن نیز از طریق توزیعهای مختلف عرضه می‌شوند.مزیت این نرم‌افزار آسان بودن نصب و به کارگیری آن است شما برای نصب و نحوه پیکربندی آن می توانید از مستندات ارائه شده همراه آن استفاده کنید.

Calamaris

Calamaris نیز یکی دیگر از این ابزارها است که برخلاف Sarg که درواقع نتایج فعالیت‌های کاربران را به طور مستقیم ارائه می کند این ابزار با بررسی و آنالیز access.log به بررسی آمارهای جالب در مورد فعالیت کاربران در طی یک زمان معین می‌پردازد به طور مثال این نرم‌افزار قادر است با بررسی آدرس‌های جستجو شده به وسیله کاربران آماری از پربازدیدترین وب سایتهای مشاهده شده به وسیله کاربران شبکه را نشان دهد که البته sarg نیز این قابلیت را داراست همچنین بررسی دامنه های مختلف و ارائه پرطرفدارترین دامنه ،ارائه آمارهای جالب در مورد خطاهای ایجاد شده در شبکه ،محبوبترین محتوای مورد علاقه کاربران و بسیاری از اطلاعات دیگر.به طور کلی Calamaris یک ابزار آنالیز تمام‌عیار است که تقریباً تمامی خصوصیات یک آنالیزور کامل را داراست و اطلاعات بسیار جامعی را در مورد فعالیتهای کاربران در شبکه به دست می‌دهد .نحوه نصب و راه اندازی آن بسیار آسان است و بسته های آماده آن به وسیله توزیعهای مختلف لینوکس ارائه می شود به هر حال برای نصب آن در توزیعهای ردهت و توزیعهای مبتنی بر آن از دستور yum install calamaris و در توزیعهای مبتنی بر دبیان از دستور apt-get install calamaris استفاده کنید.

ابزارهای متنوعی در این باره وجود دارد که هر کدام از آن‌ها خصوصیات منحصر به فرد خود را دارا هستند جهت کسب اطلاعات بیشتر در این باره و همچنین مشاهده فهرست کامل این ابزارهای به آدرس رسمی آن یعنی http://www.squid-cache.org/Misc/log-analysis.html مراجعه نمائید.

خلاصه فصل

تمامی مباحث این فصل به چگونگی نصب و راه اندازی رابط گرافیکی Cache Manager اختصاص یافته است به طور کلی مطالب زیر در این فصل پوشش داده شده اند:

- نصب و پیکربندی وب سرور Apache به عنوان یکی از پیش نیازهای Cache manager
- نحوه پیکربندی اسکوئید به منظور کسب مجوز جهت استافده از Cache manager
- بررسی و ارائه مثال از برخی گزینه های موجود درCache manager

هم اکنون در پایان این فصل شما باید بتوانید رابط Cache manager را به صورت صحیح برروی سرور نصب و راه اندازی نمائید.

فصل۷

افزایش امنیت شبکه از طریق ابزارهای احراز هویت در اسکوئید

در فصل قبل در مورد چگونگی نصب ، راه اندازی و استفاده از رابط گرافیکی مدیریت اسکوئید ، آنالیز فایلهای ثبت رویداد به منظور بررسی سطح عملکرد اسکوئید مطالب مفیدی را آموختیم.در این فصل مطالب مختلفی را در مورد استفده از ابزارهای تشخیص هویت به مراه اسکوئید به منطور افزایش سطح امنیت پراکسی سرور و جلوگیری از دسترسی کاربران غیر مجاز و همچنین چگونگی ساخت ابزارهای تشخیص هویت به صورت کاملا سفارشی و مطابق نیازهای خود خواهیم آموخت.

به طو رکلی مطالب مورد بحث در این فصل به صورت زیر است:

- احراز هویت در اسکوئید
- احراز هویت کاربران از طریق پروتکل HTTP
- احراز هویت از نوع digest
- احراز هویت از طریق Microsoft NTLM
- احراز هویت از نوع Negotate
- استفاده از چند ابزارها احراز هویت به صورت همزمان
- چگونگی نوشتن ابزارهای احراز هویت به صورت سفارشی
- آشنایی با مشکلات رایج در احراز هویت

احراز هویت از طریق پروتکل HTTP

تا کنون در فصل های گذشته در مورد راههای مختلف کنترل دسترسی به پراکسی سرور از طریق اعمال کنترل بر ip ها و MAC آدرس ها مطالب مفصلی را بیان کردیم اما با وجود تمامی این تدابیر باز هم افرادی هستند که به سادگی می توانند این محدودیت ها را از بین برده و به پراکسی سرور شما به طور غیر مجاز دسترسی داشته باشند .یکی دیگر از راههای جلوگیری از دسترسی غیر مجاز ،استفاده از ابزارهای احراز هویت که برپایه نام کاربری و رمز عبور عمل می کنند می باشد با وجود این ابزارها که از امنیت خوبی برخوردارند امنیت شبکه شما تا حد زیادی تامین خواهد شد به طوریکه فقط کاربرانی که از نام کاربری و رمز عبور معتبر برخوردار باشند می تواند به پراکسی سرور دسترسی داشته باشند.

ابزارهای احراز هویت اسکوئید از مکانیسم ساده ای بهره می برند به طوریکه از طریق هدر useragent و یا مرورگر سرفایهای حاوی پیام تصدیق هویت را به سوی کاربران می فرستد که در این پیام ها تمامی اطلاعات شامل نامهای کاربری و رمز عبور به صورت رمزنگاری شده فرستاده شده و در مقابل پس از تکمیل اطلاعات توسط کاربر و ارسال آن به سوی پراکسی سرور ، اسکوئید اطلاعات رمزگذاری شده را باز کرده و با اطلاعات موجود در پایگاه داده خود تطبیق می دهد و در صورتکیه اطلاعات ارسال شده توسط کاربر با آنچه که در پایگاه داده سرورپراکسی تعریف شده مطابقت داشته باشد کاربر مورد نظر مرحله احراز هویت را با موفقیت پشت سرگذاشته و می تواند از پراکسی سرور استفاده نماید اما در صورتکیه اطلاعات کاربری نادرست باشد پیام ۴۰۷ یا عدم موفقیت فرایند احراز هویت از طرف پراکسی سرور به سوی کاربر ارسال می شود این پروسه در واقع تمامی آن چیزی است که میان کاربر و پراکسی سرور روی می دهد.اسکوئید در حال حاضر از چهارنوع مختلف احراز هویت به نام های basic , ntlm , digest و negotiate پشتیبانی می کند که هر کدام دارای مزایا و معایب خاص خود هستند.ابزارهای احراز هویت اسکوئید از طریق دستور auth_param پیکربندی می شوند که این دستور خود دارای گزینه های انتخابی مختلفی است که در ادامه به معرفی هریک می پردازیم.

احراز هویت پایه (Basic)

این نوع ساده ترین شکل احراز هویت محسوب می شود که اسکوئید قادر به پشتیبانی و به کارگیری آن است. یکی از معایب بزرگ این نوع امنیت پایین آن است که دلیل آن هم استفاده از سیستم رمزنگاری ۶۴ بیتی است که به راحتی می توان آن را رمزگشایی کرده و به اطلاعات موجود در آن دسترسی پیدا کرد. به طورکلی استفاده از این نوع سیستم احراز هویت توصیه نمی شود زیرا هر فردی که بتواند بر داده های مبادله شده در شبکه دسترسی داشته باشد می تواند این نوع رمز نگاری را گشوده و به اطلاعات مبادله شده میان کاربر و سرور دسترسی پیدا کند. در مقابل سایر ابزارهای موجود مانند digest و negotiate توصیه می شوند.استفاده از این نوع در شبکه های کوچک و ایزوله شده که احتمال خرابکاری در آنها کمتر است به دلیل پیکربندی آسان و سریع آن نسبت به سایر انواع توصیه می شود. دستورات پیکربندی نوع basic به طور کلی به صورت زیر

است:

```
auth_param basic program COMMAND
auth_param basic utf8 on|off
auth_param basic children NUMBER [startup=N] [idle=N] [concurrency=N]
auth_param basic realm STRING
auth_param basic credentialsttl TIME_TO_LIVE
auth_param basic casesensitive on|off
```

گزینه program مسیر فایلهای اجرایی را نشان می دهد همچنین علاوه بر این می توان آرگومانهای مختلفی را به آن اضافه کرد مسیر پیش فرض این فایلها در ${prefix}/libexec/ قرار دارد که مسیر ${prefix} هنگام کامپایل اسکوئیداز طریق پیشوند prefix-- قابل تغییر است.کد مزبور فرایند احراز هویت کاربر را انجام می دهد به طوریکه اگر نام کاربری و رمز عبور ارائه شده از سوی کاربر معتبر باید خروجی برنامه OK و در غیر اینصورت ERR خواهد بود.پارامتر utf8 یکی از فرمتهای encode کردن متون است که مواردی همچون ارسال پیام ها به زبانی غیر از لاتین به کار می رود مثلا در صورتکیه پیام ارسالی به کاربر از طریق مرورگر به زبان فارسی باشد این گزینه حتما باید فعال باشد تا پیام ها به صورت صحیح توسط کاربر قابل دریافت شود و متون در هم ریخته نباشد.پارامتر children تنظیمات مختلفی انجام می دهد در حالت عادی اسکوئید بیش از یک ابزار احزار هویت را به طور همزمان در حالت اجرا دارد که این تعداد کاملا به تعداد درخواستهای موجود بستگی دارد این عمل موجب می شود میزان تاخیر به وجود آمده به وسیله ارسال و دریافت پیام ها میان کاربران و سرور به پایین ترین حد ممکن برسد.NUMBER تعداد پروسه های درحال اجرا را تعیین می کند این تعداد با توجه به وسعت شبکه باید به مقدار کافی باشد زیرا تعداد کم و درخواست زیاد موجب ایجاد تاخیر در پاسخها خواهد شد البته این اطمینان وجود دارد که تعداد زیاد آن هم منابع زیادی را به کار نمی گیرند و میزان استفاده منابع آنها در حد معقول است.گزینه های idle و startup به همراه childrenتعداد پروسه های قابل اجرا هنگام راه اندازی دوباره اسکوئید را تعیین می کنند و همراه بیشترین تعداد پروسه های قابل تعریف توسط ما در دسترس خواهند بود.این گزینه ها باعث می شوند اسکوئید در مواردی که ترافیک شبکه بسیار زیاد است بتواند با کمترین تاخیر ممکن کلیه درخواست های تصدیق هویت را بررسی کرده و آنها را در کمترین زمان ممکن پاسخ دهد.گزینه concurrency تعداد درخواستهای همزمان ممکن را که یک ابزار احراز هویت می تواند بررسی و در حد امکان پاسخ دهد را تعیین می کند بسیاری از ابزارهای تشخیص هویت فقط می توانند یک درخواست را درآن واحد پاسخ دهند مقدار پیش فرض این گزینه صفر است اما در صورتکیه ابزار مورد نظر توانایی پاسخگویی به بیش از یک درخواست را دارد می توان این مقدار را به میزان مورد نظر تغییر داد. نکته قابل توجه در این مورد این است که این دستور در اسکوئید نسخه

۳.۲ به بعد پشتیبانی می شود که البته در پایان این فصل در این مورد مطالب بیشتری ارائه می شود.

گزینه realm پیام های ارائه شده به کاربر به وسیله http client را تعیین می کند. Credentialsttl میزان زمانی را که پیام فرستاده شده به کاربر را معتبر تلقی می کند را تعیین می نماید مقدار این گزینه باید به حد کافی طولانی باشد تا مطمئن باشیم کاربر این پیام را دریافت کرده و فرصت کافی برای پاسخگویی به آن را خواهد داشت و از طرفی از ارسال پی در پی درخواست به سرور به دلیل کوتاه بودن زمان جلوگیری شود. Casesensitive نیز همانطور که از نام آن می توان حدس زد ، حساسیت به بزرگ و کوچک بودن حروف را مشخص می کند در صورتکیه این گزینه فعال باشد سرور به کوچک و بزرگ بودن حروف حساس شده و میان مقادیر بزرگ و کوچک تفاوت قائل می شود توجه به این نکته ضروری است که برخی از پایگاههای داده حروف بزرگ و کوچک را یکی تلقی کرده و میان آنها تفاوتی قائل نیستند همچنین فعال یا غیر فعال بودن آن بر ACL های نوع max_user_ipتاثیر مستقیم دارد بنابراین در به کارگیری آن نهایت دقت را داشته باشید.دستورات زیر یک نمونه از فایل پیکربندی ارائه شده برای احراز هویت کاربران از طریق ابزار Basic را نشان می دهد:

```
auth_param basic program /opt/squid/libexec/basic_pam_auth
auth_param basic utf8 on
auth_param basic children 15 start=1 idle=1
auth_param basic realm Squid proxy Server at proxy.example.com
auth_param basic credentialsttl 4 hours
auth_param basic casesensitive off
acl authenticated proxy_auth REQUIRED
http_access allow authenticated
http_access deny all
```

توجه داشته باشید دستورات بالا زمانی برای اسکوئید قابل فهم هستند که قبل از آن ACL های نوع proxy_auth وجود داشته باشند در غیر اینصورت موارد بالا باعث ایجاد خطا در زمان راه اندازی خواهند شد.

پایگاه داده مربوط به ابزرهای تشخیص هویت

ابزار احراز هویت basic_db_auth قادر است مشخصات اعتباری کاربران نظیر نام کاربری و رمز عبور را از طریق اطلاعات موجود در پایگاه داده بررسی کند به طوریکه به ازای هر نام کاربری و رمز عبور یک جدول در پایگاه داده مورد نظر برای این ابزار در نظر گرفته می شود و هنگام دریافت اطلاعات کاربری ، مشخصات موجود را با اطلاعات موجود در پایگاه

تطبیق می دهد و با توجه به معتبر یا نامعتبر بودن اطلاعات ، مجوز دسترسی را برای کاربر مورد نظر صادر می کند.

پیکربندی پایگاه داده تشخیص هویت

هریک از درخواستهای دریافت شده از سوی کاربران برای اینکه یک درخواست مجاز تلقی شود می بایست اطلاعات دریافت شده توسط گزینه های خاصی که در پایگاه داده ایجاد شده اند بررسی شوند و در صورتکیه تمامی اطلاعات موجود مجاز و مورد تایید باشند آن درخواست مجاز شمرده می شود. درجدول زیر این گزینه ها را معرفی شده اند:

--dsn	این دستور برای مشخص کردن نام اصلی پایگاه داده (DSN) به کار می رود و ابزارهای احراز هویت از آن برای اتصال به پایگاه داده استفاده می کنند. مقدار پیش گزیده ی آن DBI:mysql:database=squid است که نام مورد نظر به جای عبارت squid قرار می گیرد به طور مثال در صورتکیه نام پایگاه داده خود را به netclients تغییر دهیم عبارت بالا به صورت DBI:mysql:database=netclients تغییر می یابد. البته هر دیتابیس دیگری که از این سبک نام گذاری استفاده می کند مانند SQL ها می توانند به عنوان سرور پایگاه داده حتی خارج از سرور پراکسی مورد استفاده قرار گیرند به مثال زیر توجه کنید: DBI:mysql:database=clients:example.com:3306
--user	این گزینه برای مشخص کردن نام کاربری برای اتصال به پایگاه داده به کار می رود.
--password	برای مشخص کردن رمز عبور مورد نظر هنگام اتصال به پایگاه داده از این گزینه استفاده می شود.

--table	اسکوئید از این دستور به منظور جستجوی نامهای کاربری و رمز عبور مشخص شده با استفاده از دستور --table استفاده می کند که نام پیش فرض این جدول passwd تعیین شده است.
--usercol	این دستور به منظور تعیین نام ستونهای پایگاه داده ای که با استفاده از دستور --usercol ایجاد شده اند به کار می رود و نام های کاربری را در خود نگهداری می کند همچنین مقدار پیش فرض آن user است.
--passwdcol	این دستور برای مشخص کردن نام ستون جدول های نگهداری رمزهای عبور در پایگاه داده به کار می رود و مقدار پیش فرض آن password است.
--plaintext	رمزهای عبور نگهداری شده در پایگاه داده می توانند به دو صورت متنی و یا رمز نگاری شده نگهداری شوند که ابزارهای تشخیص هویت موجود مقدار پیش فرض را رمز نگاری شده فرض می کنند. برای اینکه بتوان آنها را بصورت متنی و بدون رمز نگاری نگهداری کرد باید مقدار این گزینه را به ۱ تغییر داد.
--cond	این گزینه جز تنظیمات اصلی محسوب نمی شود و به طور کلی برای اعمال محدودیت برروی برخی از کاربران به کار می رود مثلا در صورتکیه بخواهیم برخی از نامهای کاربری موجود در پایگاه داده را به صورت deny در آوریم به کار می رود. برا ی اینکه بتوانیم محدودیت های خاصی را اعمال کنیم عبارت cond --is enabled=1 را به کار می بریم که معنی آن ایجاد یک حالت خاص در پایگاه داده است در غیر اینصورت به جای مقدار یک ،مقدار خالی "" قرار می دهیم.
--md5	یکی از پروتکل های رمزنگاری است که می توان همراه با

رمزهای عبور به کار برد.	
یکی دیگر از تنظیمات پایگاه داده که سطح کاربرد آن در مورد رمزهای عبور است که در واقع به منظور افزایش امنیت داده ها به کار می رود.	--salt
در صورت به کارگیری این دستور اتصالات ایجاد شده به پایگاه داده حتی با وجود اتصالات جدید به صورت موازی برقرار خواهند ماند.	--persist
اگر پایگاه داده ای از نوع جوملا در اختیار داشته باشید با استفاده از این گزینه می توان آن را به ابزار احراز هویت مورد استفاده ی خود معرفی کنید.برای کسب اطلاعات بیشتر دراین باره به نشانی www.joomla.org مراجعه نمائید.	--joomal

بنابراین با توجه به تعاریف بالا تنظیمات پیکربندی برای ابزار basic_db_auth به صورت زیر است:

```
auth_param basic program /opt/squid/libexec/basic_db_auth --dsn "DBI:
mysql:database=squid_auth" --user 'db_squid' --password 'sQu1Dp4sS' --
table 'clients'--cond ''
```

این خطوط ابزاراحراز هویت اسکوئید را از نوع basic در نظر گرفته و تنظیمات مختلفی را که در جدول بالا توضیح داده شدند به آن افزوده است.

ابزار NCSA

NCSA یکی دیگر از ابزارهای تشخیص هویت است که می توان آن را همراه با اسکوئید به کار برد.راه اندازی وپیکربندی این ابزار آسان بوده و تمام کاری که باید انجام شود آن است که نامهای کاربری و رمز عبور در یک فایل با فرمت مشخص ذخیره کرده و آن را همراه با NCSA به کار ببریم. برای ساخت و مدیریت نامهای کاربری می توان از htpasswd که یکی از اجزای وب سرور آپاچی است استفاده کرد.فرض کنیم رمزهای عبور تعیین شده در مسیر/opt/squid/etc/passwd

نگهداری می شوند ، در این صورت نام های کاربری و رمز عبور های تعیین شده به صورت زیر به آن اضافه می شوند:

```
htpasswd /opt/squid/etc/passwd saini

New password:

Re-type new password:
```

شکل کلی اضافه کردن رمز عبور به صورت بالا است . اکنون باید فایل ایجاد شده را به NCSA معرفی کنیم که با توجه به مثال بالانحوه پیکربندی آن به صورت زیر است:

```
auth_param basic program /opt/squid/libexec/basic_ncsa_auth/opt/
squid/etc/passwd
```

همانطور که گفته شد مزیت اصلی این برنامه عدم نیاز به پایگاه داده و پیکریندی سریع وآسان آن می باشد که با انجام مراحل بالا به راحتی این ابزار پیکربندی شده و آماده ی به کارگیری همراه با اسکوئید است.

ابزار NIS

NISیکی دیگر از ابزارهای احراز هویت موجود است که توسط شرکت سان مایکروسیستمز ایجاد شده است و مخفف عبارت Network Information Service است. برای اینکه بتوان از این ابزار همراه با اسکوئید استفاده کرد کافی است آدرس دامنه و رمز عبور پایگاه داده ای که قبلا ایجاد کرده ایم را به صورت زیر به تنظیمات پیکربندی این ابزار اضافه کنیم:

```
auth_param basic program /opt/squid/libexec/basic_nis_auth example.com
passwd.byname
```

ابزار LDAP

LDAP یا (Lightweight Directory Access Protocol) و (basic_ldap_auth) که همراه با یکدیگر سرور LDAPرا تشکیل می دهند، یکی دیگر از ابزارهای احراز هویت است که فرایند تشخیص هویت را از طریق سرورهای LDAP انجام می دهد. برای این کار کتابخانه های توسعه ی LDAP باید در سرور مورد نظر وجود داشته باشند جهت کسب اطلاعات بیشتر در مورد نحوه نصب و آشنایی با پیکربندی آن به آدرس http://www.openldap.org مراجعه کنید. basic_ldap_auth شامل گزینه های زیادی برای پیکربندی و تغییر شرایط احراز هویت است که در اینجا فقط به نحوه به کارگیری آن با اسکوئید اشاره می شود همچنین جهت کسب اطلاعات جامع تر در این باره می توانید همراه عبارت basic_ldap_auth را در مستندات همراه اسکوئید جستجو نمائید.برای مثال نحوه پیکربندی اتصال آن به اسکوئید به

صورت زیر است:

```
auth_param basic program /opt/squid/libexec/basic_ldap_auth -b
"dc=example,dc=com" ldap.example.com
```

در این تنظیمات ، ldap.example.com نام سرور LDAP مورد نظر است و نام دامنه example.com نام اختصاصی شناسایی (DN) می باشد.

ابزار SMB

ابزار SMB یا basic_smb_auth ساده ترین ابزار احراز هویتی است که می توان از آن همراه با کاربران ویندوزی و Samba به کار گرفت. برای اینکه بتوان از ابزار basic_smb_auth استفاده کنیم باید سرویس دهنده Samba برروی سروری که اسکوئید برروی آن قرار دارد و یا سروری که اسکوئید به آن دسترسی دارد وجود داشته باشد. خوشبختانه Samba به هبسیاری از توزیع های لینوکس و یونیکس پشتیبانی می شود و مستندات کاملی در وب سایت رسمی پروژه به آدرس http://www.samba.org وجود دارد که شما را در مورد نحوه نصب و پیکربندی آن راهنمایی می کند. در صورتکیه تمامی ابزارها و کتابخانه های لازم به درستی نصب و اجرا شده باشند با استفاده از کد زیر می توان از ابزار SMB استفاده نمود:

```
auth_param basic program /opt/squid/libexec/basic_smb_auth -W  WORKGROUP
```

در اینجا پسوند –W برای مشخص کردن نام دامنه ی ویندوز به کار می رود.

ابزار PAM

Pluggable Authentication Modules، PAM)) به مکانیسمی اطلاق می شود که از آن برای اجتماع گروهی از ابزارهای احراز هویت سطح پایین نظیر fingerprint ها و smartcard ها و غیره و تبدیل آنها به API ها به کار می رود .نکته ای که باید به آن توجه داشت این است که PAM ها در توزیع های BSD پشتیبانی نمی شوند اسکوئید از basic_pam_auth به منظور به کارگیری PAM استفاده می کند به هر حال برای استفاده از این ابزار باید تنظیماتی را در قسمت /etc/pam.d/ انجام داد و ماژول های مورد استفاده را در آن پیکربندی نمود.یک مثال ساده برای پیکربندی اسکوئید در دایرکتوری /etc/pam.d/ به صورت زیر است:

```
#%PAM-1.0
auth    include   password-auth
```

```
account include   password-auth
```

پس از اینکه تنظیمات مورد نظر در دایرکتوری /etc/pam.d/ برروی اسکوئید صورت گرفت باید اسکوئید را جهت استفاده از پایگاه داده PAM پیکربندی کنیم که یک نمونه از تنظیمات آن به صورت زیر خواهد بود:

```
auth_param basic program /opt/squid/libexec/basic_pam_auth
```

جهت کسب اطلاعات بیشتر در مورد تنظیمات قابل استفاده در این باره به مستندات basic_pam_auth رجوع شود.

ابزار MSNT

ابزار MSNT به منظور احراز هویت کاربران از طریق کنترل کننده ی دامنه ها در ویندوز و سامبا به کار می رود.راه اندازی و پیکربندی MSTN بسیار آسان بوده و کافی است فایل تنظیمات اصلی آن در مسیر opt/squid/etc/msntauth.conf/ را گشوده و تنظیمات مورد نظر خود را به آن اضافه نمائید.تنظیمات پیش فرض موجود در اینمسیر به صورت زیر است:

```
# NT domain hosts. Best to put the hostnames in /etc/hosts.
server myPDC        myBDC        myNTdomain
# Denied and allowed users. Comment these if not needed.
denyusers      /opt/squid/etc/msntauth.denyusers
allowusers     /opt/squid/etc/msntauth.allowusers
```

این تنظیمات پیش فرض بوده و برخی از پارامتر های کلیدی آن حتما باید با توجه به تنظیمات شما تغییر یابند که در اینجا myNTdomain ، myBDC (Backup Domain Controlle)،myPDC (Primary Domain Controller) (Windows NT Domain) از جمله پارامترهای قابل تغییر هستند همچنین در اینجا می توان به طور همزمان پنج دامنه متفاوت را به تنظیمات پیکربندی اضافه نمود. denyusers نامهای کاربری ای را که به هیچ عنوان اجازه ی دسترسی به پراکسی سرور را ندارند را مشخص می کند درواقع نام های کاربری موجود در این فایل توسط این ابزار اعتبار سنجی نمی شوند. دستور allowusers نامهای کاربری ای را که همواره اجازه ی دسترسی به پراکسی سرور را دارند را مشخص می کند توجه داشته باشید نام های کاربری تعیین شده در این دستور همواره اجازه ی دسترسی به پراکسی سرور را خواهند داشت حتی اگر فرایند اعتبار سنجی آنها ناموفق باشد .تا اینجا عمل پیکربندی MSNT پایان می یابد و برای اینکه اسکوئید بتواند با این ابزار ارتباط برقرار کند خط زیر را به تنظیمات اسکوئید اضافه می کنیم:

auth_param basic program /opt/squid/libexec/msnt_auth

ابزار MSNT multi domain

MSNT multi domain عملکردی مشابه ابزار تخیص هویت MSNT دارد با این تفاوت که کاربران قبل از وارد کردن نام کاربری خود باید نام دامنه ای که در آن عضویت دارند را وارد کنند مانند : workgroup\sarah

نحوه پیکربندی آن کاملامشابه با MSNT است و فقط کافی است خط زیر را به فایل پیکربندی اسکوئید اضافه کنید:

auth_param basic program /opt/squid/libexec/basic_msnt_multi_domain_auth

نکته: در صورت استفاده از ابزار باید از نصب بودن بسته ی **Authen::SMB** *و* **nmblookup** *و* **smbclient** *برروی سرور و همچنین فعال بودن سرویس دهنده سامبا برروی سیستم هایی که به نوعی به اسکوئید دسترسی دارند اطمینان حاصل نمائید .*

ابزار SASL

Simple Authentication and Secure Layer(SASL) یکی دیگر از ابزارهای موجود جهت احراز هویت کاربران است که مکانیسمی شبیه ابزارPAMدارد.برای پیکربندیSASLباید یک فایل پیکربندی به نام basic_sasl_auth.conf که عبارت pwcheck_method:sasldb در آن ذخیره شده باشدایجاد کنیم.پس از ایجاد فایل مورد نظر آن را در داخل دایرکتوری /usr/lib/sasl2/ کپی نمائید.و بالاخره خط زیر را به تنظیمات پیکربندی اسکوئید اضافه کنید:

auth_param basic program /opt/squid/libexec/basic_sasl_auth

نکته: برای راه اندازی صحیح این ابزار باید کتابخانه ی **Cyrus SASL** *برروی سرور وجود داشته باشد .برای دریافت آن به نشانی (***http://asg.web.cmu.edu/sasl***) مراجعه کنید.*

ابزار getpwnam

از getpwnam می توان جهت تائید احراز هویت کاربران محلی شبکه استفاده کرد این نرم افزار از ابزار getpwnam () موجود در توزیعهای یونیکس به منظور جستجوی کاربران وارد شده به سرور اسکوئید استفاده می کند به علاوه از getpwnam می توان برای تشخیص هویت کاربران از پایگاه داده ابزارهایی چون LDAP،PAM استفاده شود.تنظیمات

پیکربندی مورد نیاز جهت اتصال به اسکوئید مانند خط زیر است:

```
auth_param basic program /opt/squid/libexec/basic_getpwnam_auth
```

ابزار POP3

یکی از ابزارهای تشخیص هویت مهمی که اسکوئید به خوبی از آن پشتیبانی می کند POP3 (post office protocol v3) با استفاده از ابزار basic_pop3_auth است.برای اینکه بتوانیم این ابزار را همراه با اسکوئید به کار ببریم باید آدرس ip یا دامنه سروری که سرویس pop3 برروی آن فعال است را داشته باشیم و خط زیر را به تنظیمات اصلی اسکوئید اضافه کنیم:

```
auth_param basic program /opt/squid/libexec/basic_pop3_auth pop3.example.com
```

نکته: ابزار **basic_pop3_auth** *از بسته ی نرم افزاری* **Net::POP3** *استفاده می کند بنابراین قبل از راه اندازی آن از وجود این بسته برروی سرور اطمینان حاصل نمائید.*

ابزار RADIUS

ابزار basic_radius_auth باعث اتصال اسکوئید به یک radius سرور می شود و از این طریق می تواند کاربران را از طریق نام کاربری و رمز عبور تایید هویت نماید.خط زیر باعث ایجاد اتصال میان اسکوئید و سرور radius می شود:

```
auth_param basic program /opt/squid/libexec/basic_radius_auth -h radius.example.com
-p 1645 -i squid_proxy -w s3cR37 -t 15
```

در این خط از دستورات ،h- برای مشخص کردن آدرس اتصال به سرور به کار می رود که در اینجا radius.example.com آدرس سرور radius مورد نظر است ،p- شماره ی درگاه ارتباطی مورد نظر را مشخص می کند که در این مثال 1645 تعیین شده،i- به منظور تعیین یک شناسه منحصر به فرد که سرور radius سرور پراکسی را به این نام شناسایی می کند در صورتکیه این نام تعیین نشود از آدرس ip پراکسی سرور به عنوان نام شناسایی استفاده می شود همچنین -w کلمه ی رمز میان radius server و پراکسی سرور را تعیین می کند که این کلمه ی رمز به منظور افزایش امنیت در برقراری ارتباط میان دو سرور به کار می رود و در نهایت t- نیز میزان زمان تاخیر در ارتباطات را تعیین می کند که مقدار پیش فرض آن ۱۰ ثانیه است. در صورتکیه بخواهیم اطلاعات مربوط به سرور radius در جایی غیر از فایل پیکربندی اسکوئید

نگهداری شود می توان اطلاعات مورد نظر را در یک فایل متنی مثلا در مسیر /opt/squid/etc/basic_radius_auth.conf ذخیره کرده و اطلاعات مربوط به سرور را به صورت زیر در آن ذخیره می کنیم:

```
server radius.example.com
port 1645
identifier squid_proxy
secret s3cR37
```

اکنون تنظیمات مربوط به پیکربندی اسکوئید به صورت زیر تغییر می کند:

```
auth_param basic program /opt/squid/libexec/basic_radius_auth -f /opt
squid/etc/basic_radius_auth.conf -t 15
```

در اینجا f- مسیر فایل تنظیمات سرور radius را که basic_radius_auth جهت ارتباط با سرور Radius استفاده می کند را مشخص می کند.

ابزار Fake Basic

یکی از ابزارهای تشخیص هویت اسکوئید از نوع Fake authentication است .نکته ی جالب در باره ی این نوع این است که این ابزار اعتبار کاربران را بدون احراز هویت واقعی و جستجو در هر گونه پایگاه داده ای تایید می کند. به بیان ساده تر این نوع یک ابزار تشخیص هویت واقعی محسوب نمی شود و تمامی کاربران با هر نام کاربر ی و رمز عبوری اعتبار آنها تایید می شود. این ابزار هیچ گونه تاثیری در افزایش امنیت ندارد به طور عمده به منظور آزمایش روش های مختلف و به کارگیری آن با سایر ابزارها به صورت تلفیقی به کار می رود.

ابزار Digest

ابزار Digestیکی دیگر از انواع ابزارهای احراز هویت است که نسبت به نوع Basic از امنیت بالاتر و قابلیت های بهتری برخوردار است در این نوع برخلاف نوع Basic که نام کاربری و رمز عبور از طریق شبکه میان کاربر و سرور مبادله می شود در این نوع این عمل صورت نمی گیرد.در این نوع اطلاعات محرمانه و حساس با استفاده از پروتکل رمزنگاری md5 به صورت کد در می آید..تنظیمات کلی auth_param برای این نوع به صورت زیر است:

```
auth_param digest program COMMAND
```

```
auth_param digest utf8 on|off
auth_param digest children NUMBER [startup=N] [idle=N] [concurrency=N]
auth_param digest realm STRING
auth_param digest nonce_garbage_interval TIME
auth_param digest nonce_max_duration TIME
auth_param digest nonce_max_count NUMBER
auth_param digest nonce_strictness on|off
auth_param digest check_nonce_count on|off
auth_param digest post_workaround on|off
```

پارامترهای program, utf8, children و realm با همان معنایی که در نوع BASIC دارند در این نوع نیز به کار می روند.در جدول زیر سایر پارامترهای موجود در نوع DIGEST شرح داده است:

پارامتر مورد نظر	توضیحات
nonce_garbage_ interval	مندت زمانی که یک پیام تشخیص هویت که به سوی کاربر ارسال شده است دوباره اعتبار سنجی شود.
nonce_max_duration	مدت زمانی که یک پیام اعتبار سنجی معتبر باقی می ماند.
nonce_max_count	این گزینه تعداد دفعاتی که یک پیام اعتبار سنجی معتبر باقی می ماند را مشخص می کند.
nonce_strictness	هنگامی که درخواستهای اتعبار سنجی به طور مکرر برای کاربران ارسال می شود تعدادی از این پیام ها بی پاسخ باقی می مانند و به عنوان درخواستهای بدون جواب باقی می مانند.این دستور مشخص می کند که آیا انجام چنین کاری مجاز است یا خیر که مقدار پیش فرض آن off است.

check_nonce_count	اسکوئید را به اجبار وادار به اعتبار سنجی پیام های احراز هویت می کند و در صورتکیه کاربری اعتبار سنجی نشده باشد با پیام ۴۰۱ یعنی عدم اعتبار سنجی مواجه می شود این گزینه در مواردی که برخی از افراد با هدف سواستفاده و اختلال در شبکه پیام های مکرر احراز هویت را به سوی سرور ارسال می کنند با پیام ۴۰۱ روبرو می شوند.
post_workaround	زمانی که کاربران به طور مکرر پیام های اعتبار سنجی را به سوی سرور ارسال می کنند که برخی از این کاربران به درتی اعتبار سنجی می شوند و برخی دیگر درخواستهای خود را تکرار می کنند این دستور در این گونه موارد باعث تسریع در انجام درخواستها خواهد شد.در واقع دستور post_workaround به نوعی باعث مدیریت پردازش درخواستها می شود.

با توجه به گزینه های مورداستفاده در نوع Digest خطوط زیر مثالی از تنظیمات پیکربندی اسکوئید برای این نوع را نشان می دهد:

```
auth_param digest program /opt/squid/libexec/digest_file_auth
auth_param digest utf8 on
auth_param digest children 20 startup=0 idle=1
auth_param digest relm Squid proxy server at proxy.example.com
auth_param digest nonce_garbage_interval 5 minutes
uth_param digest nonce_max_duration 30 minutes
auth_param digest nonce_max_count 50
auth_param digest nonce_strictness on
auth_param digest check_nonce_count on
```

```
auth_param digest post_workaround on
acl authenticated proxy_auth REQUIRED
http_access allow authenticated
http_access deny all
```

در ادامه به معرفی ابزارهای احراز هویت از نوع Digest می پردازیم.

ابزار File

digest_file_auth یکی از ابزارهای نوع digest است که قادر به نگهداری اطلاعات کاربران در فایلهای متنی و رمزنگاری شده از نوع md5 است. اگر نامهای کاربری و رمز عبور در فایل متنی نگهداری شود شکل کلی نگهداری اطلاعات کاربران به صورت زیر است:

```
username:password
```

در صورتکیه اطلاعات کاربران را در فایلهای متنی نگهداری کنیم به لحاظ امنیتی عمل خاصی انجام نداده ایم و تنها مزیت این نوع نسبت به انواع basic این است که اطلاعات کاربری از طریق شبکه به طور مستقیم میان کاربران و سرور مبادله نمی شود در حالیکه همچنان اطلاعات به صورت کاملا واضح در فایل متنی قابل دسترسی و استفاده است.روشی دیگر برای افزایش امنیت نگهداری رمز ورود به صورت پروتکل رمزنگاری md5 است که در این صورت شکل کلی اطلاعات به صورت username:realm:HA1 است.در این فرمت HA1از نوع md5 است.پس از انتخاب فرمت نگهداری ،فایل موجود آماده استفاده است.تنظیمات پیکربندی اسکوئید برای برقراری ارتباط با نوع file authentication به صورت زیر است:

```
auth_param digest program /opt/squid/libexec/digest_file_auth -c /opt/
squid/etc/digest_file_passwd
```

نکته ی قابل توجه در این تنظیمات فرمان c– است ،در صورتکیه اطلاعات کاربران را در فایل متنی وبدون هیچ گونه عملیات رمزنگاری نگهداری می کنیم به کارگیری این فرمان مجاز نیست و در صورتکیه اطلاعات به صورت رمزنگاری شده نگهداری می شوند استفاده از این فرمان الزامیست.

ابزار LDAP

با استفاده از دستور digest_ldap_auth می توان از ابزار تشخیص هویت LDAP در نوع digest استفاده کرد پارامترها ومکانیسم کلی آن مشابه نوع basic است که در پیش تر معرفی شد.برای اینکه بتوانیم ابزار digest_ldap_auth را به همراه اسکوئید به کار بگیریم باید خطوط زیر را به فایل پیکربندی اسکوئید اضافه کنیم:

```
auth_param digest program /opt/squid/libexec/digest_ldap_auth

-b "ou=clients,dc=example,dc=com" -u "uid" -A "l"
-D "uid=digest,dc=example,dc=com"
-W "/opt/squid/etc/digest_cred" -e -v 3 -h ldap.example.com
```

نام دستور	توضیحات
-b	این دستور (distinguished name (DN را مشخص می کند.
-u	برای مشخص کردن نام کاربری مورد نیاز در این ابزار از –u استفاده می شود.
-A	به نوعی خاصیت ایجاد رمز ورود را شناسایی وفعال می کند که مقدار آن در حالت فعال 1 است.
-D	فرایند جستجو را در DN فراهم می کند.
-W	نشان دهنده مسیر فایلی است که رمز ورود در آن نگهداری می شود.
-e	باعث رمزنگاری برروی رمز ورود می شود.
-v	این فرمان نسخه ی جاری LDAP را نشان می دهد.

آدرس سرور LDAP برای اتصال به آن را مشخص می کند که در این مثال ldap.example.com آدرس رور مورد نظر است.	-h

ابزار eDirectory

eDirectory authrntication ابزار احراز هویت اختصاصی شرکت ناول است که اسکوئید قادر به پشتیبانی از آن است.ابزار مورد استفاده همراه با اسکوئید digest_edirectory_authentication نام دارد.تنظیمات پیکربندی آن مشابه digest_ldap_auth است همچنین جهت کسب اطلاعات بیشتر به http://en.wikipedia.org/wiki/Novell_edirectory مراجعه کنید.

ابزار Microsoft NTLM

NT LAN Manager (NTLM) یکی از ابزارهای تشخیص هویت نوع digest است که توسط شرکت مایکروسافت توسعه می یابد.برخی از خصوصیات منحصر به فرد NTLM عبارتند از:

- NTLM فقط قادر به احراز هویت برروی اتصالات TCP است .
- قادر به ایجاد محدودیت برروی سرعت و ظرفیت اتصالات کاربر است.
- از نوع پروتکل باینری است بنابراین فقط windows domain controller قادر به استفاده از آن است.

اطلاعات بیشتر در مورد این ابزار در http://en.wikipedia.org/wiki/NTLM موجود است.

پارامترهای کلی auth_param که NTLM قادر به پشتیبانی از آن است عبارتند از:

auth_param ntlm program COMMAND

auth_param ntlm children NUMBER [startup=N] [idle=N] [concurrency=N]

auth_param ntlm keep_alive on|off

پارامترهای program و children عملکردی مشابه موراد موجود در انواع basic و digest که در قسمتهای قبلی شرح داده شده اند دارند همچنین در صورتکیه مقدار پارامتر keep_alive، off باشد اسکوئید درخواستهای اولیه ای که از طرف

مرورگرهای کاربران به منظور بررسی پشتیبانی از این نوع ارسال می شود را خاتمه می دهد که البته مقدار پیش فرض این دستور on است.خطوط زیرنمونه ای تنظیمات پیکربندی اسکوئید به منظور پشتیبانی از ابزار NTLM است:

```
auth_param ntlm program /opt/squid/libexec/ntlm_smb_lm_auth
auth_param ntlm children 20 startup=0 idle=1
auth_param ntlm keep_alive on
acl authenticated proxy_auth REQUIRED
http_access allow authenticated
http_access deny all
```

ابزار Samba's NTLM

در صورتکیه بخواهیم از ابزار NTLM همراه با SAMBA استفاده کنیم می توانیم از ابزار مناسب این برنامه که اتفاقا خود بخشی از سامبا محسوب می شود استفاده کنیم این ابزار ntlm_auth نام دارد.تنظیمات مورد نیاز برای این نوع به صورت زیر است:

```
auth_param ntlm program /absolute/path/to/ntlm_auth --helper-protocol=squid-2.5-ntlmssp
```

برای اینکه بتوانیم فرایند احراز هویت و ایجاد محدودیت را برروی گروههای مختلف اعمال کنیم دستور --require-membership-of را همراه با تنظیمات پیکربندی قبلی به صورت زیر به کار می بریم:

```
auth_param ntlm program /absolute/path/to/ntlm_auth
--helper-protocol=squid-2.5-ntlmssp
--require-membership-of="WORKGROUP\Domain Users"
```

دراین صورت کاربران فقط زمانی قادر به استفاده از پراکسی سرورند که یکی از اعضای گروه تعریف شده باشند.جهت دریافت مستندات کامل به http://www.samba.org/samba/docs/man/manpages/ntlm_auth.1.html

مراجعه کنید.

ابزار Fake NTLM

مشابه basic_fake_auth،ntlm_fake_auth نیز فرایند احراز هویت را به طور واقعی انجام نمی دهد وتمامی کاربران با هر نام کاربری ای قادر به استفاده از پرکسی سرور هستند این نوع همانطور که از نامش پیداست ،ابزار احراز هویت واقعی محسوب نمی شود و بیشتر برای تست عملکرد ابزار تشخیص هویت به کار می رود و هیچ گونه تاثیری بر امنیت ندارد.

ابزار Negotiate

ابزار Negotiateیکی دیگر از انواع ابزارهای تشخیص هویت است که قابلیت های منحصر به فرد خود را داراست همچنین این نوع از سازگاری بیشتری با سرویس Active Directory برخوردار است و به طور کلی در این نوع به مقوله ی امنیت بیشتر پرداخته شده است و نسبت به انواع قبلی از امنیت به نسبت مراتب بالاتری برخوردار است. بزار negotiate_kerberos_auth به منظور ارتباط با اسکوئید در نظر گرفته شده است.مراحل راه اندازی و پیکربندی این نوع شامل چهار مرحله ی کلی است به شرح زیر است:

- قبل از هر چیز،باید فایل keytab را از طریق ابزار ktpass که در سیستم عامل ویندوز موجود است ایجاد کنیم:

```
ktpass -princ HTTP/proxy.example.com@REALM
-mapuser proxy.example.com -crypto rc4-hmac-nt pass s3cr3t
-ptype KRB5_NT_SRV_HST -out squid.keytab
```

قبل از اجرای دستورات بالا باید یک حساب کاربری به نام proxy.example.com در سیستم عامل ویندوز ایجاد کنیم پس از آنکه فایل keytab ساخته شد باید آنرا به یک مکان مناسب در سرور اسکوئید مثلا opt/squid/etc/squid.keytab/ انتقال دهیم همچنین باید مطمئن شویم که فقط اسکوئید می تواند به این فایل مهم دسترسی داشته باشد.

- پس از اتمام مرحله ی اول نوبت به پیکربندی پروتکل Kerberos که یک پروتکل احرازبه شما رمی رود می رسدبرای اینکه بتوان از Kerberos همراه با اسکوئید استفاده کرد باید فایل etc/krb5.conf/ را به صورت زیر ویرایش کرد:

```
[libdefaults]
 default_realm = REALM
```

```
dns_lookup_realm = true
dns_lookup_kdc = true
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true
```

- در این مرحله باید تغییراتی را در فایل startup اسکوئید ایجاد کنیم قبل از هر چیز مطالب فصل گذشته در مورد راه اندازی اسکوئید را مطالعه نمائید.برای این کار باید خط زیر را به اسکریپت startup اضافه کنیم:

```
export KRB5_KTNAME=/etc/squid/squid.keytab
```

- و در نهایت خطوط زیر را به فایل پیکربندی اسکوئید اضافه می کنیم:

```
auth_param negotiate program /opt/squid/libexec/negotiate_
kerberos_auth
auth_param negotiate children 15
auth_param negotiate keep_alive on
acl authenticated proxy_auth REQUIRED
http_access allow authenticated
http_access deny all
```

استفاده از چند ابزار تشخیص هویت به طور همزمان

با استفاده از دستور auth_param می توان اسکوئید را مجاب به استفاده از چند ابزار تشخیص هویت به صورت همزمان نمود.برای این کار کافی است برای هر یک از ابزارها دستور auth_param را به کار ببریم.هنگام استفاده از چند ابزار ، اسکوئید فهرستی از ابزرهای احراز هویت را برای کاربران نمایش می دهد با توجه به توضیحات موجود در نشانی http://www.ietf.org/rfc/rfc2617 ، هر کاربر می بایست قویترین ابزار را از دیدگاه خود انتخاب نماید.بنابراین هنگام پیکربندی اسکوئید باید قویترین و مطمئن ترین ابزارها را به ترتیب پیکربندی کنیم:

۱. ابزار Negotiate/Kerberos

۲. ابزار Microsoft NTLM

۳. ابزار Digest

۴. ابزار Basic

این فهرست به صورت پیشنهادی مطرح شده است و هیچ گونه اجباری در انتخاب انواع ابزارها وجود ندارد بنابراین شما با توجه به موقعیت شبکه ی خود می توانید انواع مناسب را انتخاب کرده و به کار ببرید.

ایجاد یک ابزار تشخیص هویت به صورت سفارشی

تا کنون ابزارهای مختلفی را در حوزه ی احراز هویت کاربران معرفی کردیم که هر کدام از آنها بسته به خصوصیات خود دارای مزایا و معایبی بودند ،سطح امنیت درآنها متفاوت بوده همچنین راه اندازی و پیکربندی برخی از آنها زمانبر و با دشواریهایی روبرو است.یکی از راههای ساده برای رهایی از بعضی مشکلات و محدودیت ها امکان طراحی و ایجاد ابزار احراز هویت با توجه به انتظارات و قابلیتهای مورد نظر خود است. دستورات زیر یک نمونه از این ابزارهاست که به زبان پیتون طراحی شده است و امکان تشخیص هویت کاربران را در سطح امنیت پایه برای ما فراهم می کند البته شما با توجه به نیازهای خود می توانید در کدهای زیر تغییراتی ایجاد کرده و آن را مطابق سلیقه ی خود طراحی کنید:

```
#!/usr/bin/env python
import sys
 def validate_credentials(username, password):  """
  Returns True if the username and password are valid.  False otherwise  """
  # Write your own function definition.
  # Use mysql, files, /etc/passwd or some service you
  return False
if __name__ == '__main__':
  while True:
    # Read a line from stdin
    line = sys.stdin.readline()
    # Remove '\n' from line
```

```
    line = line.strip()
    # Check if string contains two entities
    parts = line.split(' ', 1)
    if len(parts) == 2:
        # Extract username and password from line
        username, password = parts
        # Check if username and password are valid
        if validate_credentials(username, password):
            sys.stdout.write('OK\n')
        else:
            sys.stdout.write('ERR Wrong username or pa
    else:
        # Unexpected input
        sys.stdout.write('ERR Invalid input\n')
            # Flush the output to stdout.
    sys.stdout.flush()
```

با ذخیره ی کدهای بالادر یک فایل متنی وانتقال آن به دایرکتوری opt/squid/libexec واضافه کردن دستورات زیر در تنظیمات پیکربندی اسکوئید می توانید از یک ابزار تشخیص هویت کاملا سفارشی شده استفاده کنید:

```
auth_param basic program /opt/squid/libexec/basic_generic_auth.py
auth_param basic children 15 startup=0 idle=1
auth_param basic realm Squid proxy server at proxy.example.com
auth_param basic credentialsttl 4 hours
auth_param basic casesensitive on
acl authenticated proxy_auth REQUIRED
```

```
http_access allow authenticated
http_access deny all
```

برخی از مسائل و مشکلات رایج

همواره در راه اندازی و پیکربندی ابزارهای احراز هویت به دلیل اعمال تنظیمات نادرست و عدم آگاهی از منشا ایجاد آن مشکلات ناخواسته ای به وجود می آید که ما دراینجا به برخی از رایج ترین این مشکلات اشاره می کنیم:

دسترسی به برخی وب سایتها بدون نیاز به احراز هویت

با توجه به محیط کاری موجود ،میخواهم برخی از وب سایتهایی مربوط به شبکه داخلی شرکت یا سازمان به طور عموم در دسترس همه ی کاربران قرار گرفته برای دسترسی به آنها هیچ گونه احراز هویتی نیاز نباشد برای اینکار می توان با ایجاد ACL های مخصوصی برای این وب گاهها استثنا قائل شد.دستورات زیر نمونه ای از این ACL ها را نشان می دهد:

```
acl whitelisted dstdomain .example.com .news.example.net
acl authenticated proxy_auth REQUIRED
# Allow access to whitelisted websites.
# But only from our local network.
# localnet is default ACL list provided by Squid.
http_access allow localnet whitelisted
# Allow access to authenticated users.
http_access allow authenticated
# Deny access to everyone else.
http_access deny all
```

د راین تظیمات همه ی کاربران موجوددر شبکه ی LAN می توانند بدون هیچ گونه محدودیتی تمامی وب سایتهای موجود در ACL ی به نام whitelisted مشاهده کنند.

احراز هویت در حالت پراکسی سرور شفاف

یکی از سوالات مهمی که در این زمینه پیش می آید امکان استفاده از ابزارهای احراز هویت در حالت intercept یا شفاف است به طور کلی به دلیل نوع ارتباطی که در حالت شفاف میان کاربران و سرور برقرار می شود استفاده از ابزارهای احراز هویت در این نوع امکان پذیر نخواهد بود.

خلاصه فصل

در این فصل انواع راههای مختلف احراز هویت کاربران معرفی وبررسی شدند همچنین انواع ابزارهای تشخیص هویت به همراه تنظیمات کلی پیکربندی آنها بررسی و بحث شدند.مطالب پوشش داده شده در این فصل شامل موارد زیر است:

- راههای مختلف احراز هویت کاربران از طریق HTTP Basic authentication
- HTTP Digest authentication و ابزارهای موجود در این نوع
- Microsoft NTLM authentication
- Negotiate authentication
- امکان ایجاد ابزارهای احراز هویت سفارشی سازی شده به وسیله مدیران شبکه مطابق با طح نیازها و انتظارات خود.

تمام مطالب این فصل به بررسی و معرفی بهترین ابزارهای احراز هویت که به همراه اسکوئید عرضه می شوند اختصاص دارد بالطبع با توجه به پیچیدگی ها و نکات ظریفی که در نحوه پیکربندی این ابزارها وجود دارد انتخاب بهترین و مناسب ترین ابزار مستلزم شناخت کافی از نیازهای شبکه همچون طح امنیت،انتظارات کاربران و ... دارد.

فصل هشتم

راه اندازی و ارتباط گروهی از کش سرورها با یکدیگر

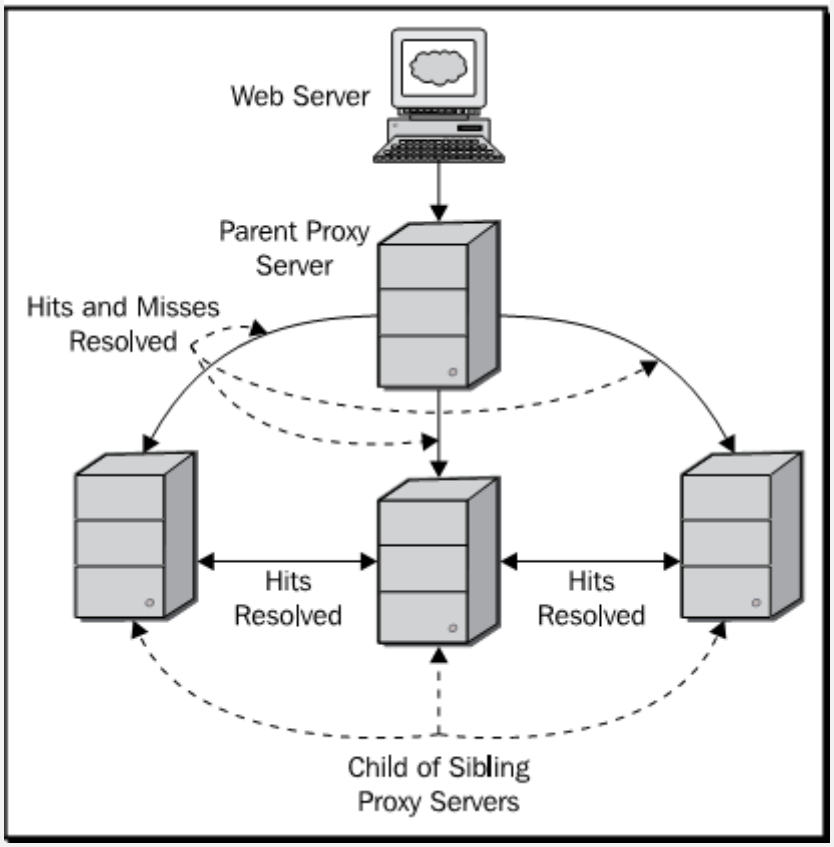
در فصل های پیشین به چگونگی ارتباط کش سرورها با یکدیگر ، هدف از ارتباط سرورها بایکدیگرو همچنین مزایا آنها مطالبی را بیان کردیم در این فصل به بررسی جزئیات و چگونگی ارتباط این سرور ها با یکدیگر مطالب مفصلی را بیان خواهیم کرد.

مطالبی که در این فصل مطرح می شوند شامل موارد زیر است:

- آشنایی با ارتباط کش سرورها با یکدیگر
- دلایل استفاده از کش سرورها به صورت گروهی
- مشکلات استفاده از کش سرورها به صورت گروهی
- تنظیمات مربوط به اسکوئید
- کنترل ارتباطات با سرورهای همسایه
- پروتکل های ارتباطی مربوط به اتصال کش سرورها با یکدیگر

ارتباطات سلسله مراتبی (گروهی)

این نوع ارتباطات به گروهی از کش سرور های موجود در یک شبکه گفته می‌شوند که با یکدیگر در ارتباط و تعامل هستند.ساختار کلی این ارتباطات مانند شاخه‌های یک درخت است که به هم متصل هستند.نوع ارتباطات موجود می تواند از نوع sibiling یا parent-child باشد هدف از این نوع ارتباطات کاهش میزان مراجعه به سرورهای مقصد و استفاده از منابع سایر سرورهای شبکه برای پاسخگویی به درخواستهای کاربران است که این عمل موجب تسریع در پاسخگویی به درخواستها ،صرفه جویی در پهنای باند ،کاهش میزان بارترافیکی بر روی وب سرور ها و در نهایت بهره گیری از حداکثر توان منابع شبکه از جمله دلایل استفاده از این نوع ارتباطات است.تصویر زیر شکل کلی این نوع ارتباطات را نشان می دهد:



تصویر ۸-۱ چگونگی ارتباط کش سرورهای مختلف با یکدیگر را نشان می دهد.

به دلیل ارتباطات گسترده ای که میان سرورهای شبکه پدید می‌آید انواع پروتکل های شبکه ای نظیر ICP, HTCP, Cache Digestsو CARP در این ارتباطات مورد استفاده قرار می گیرند.

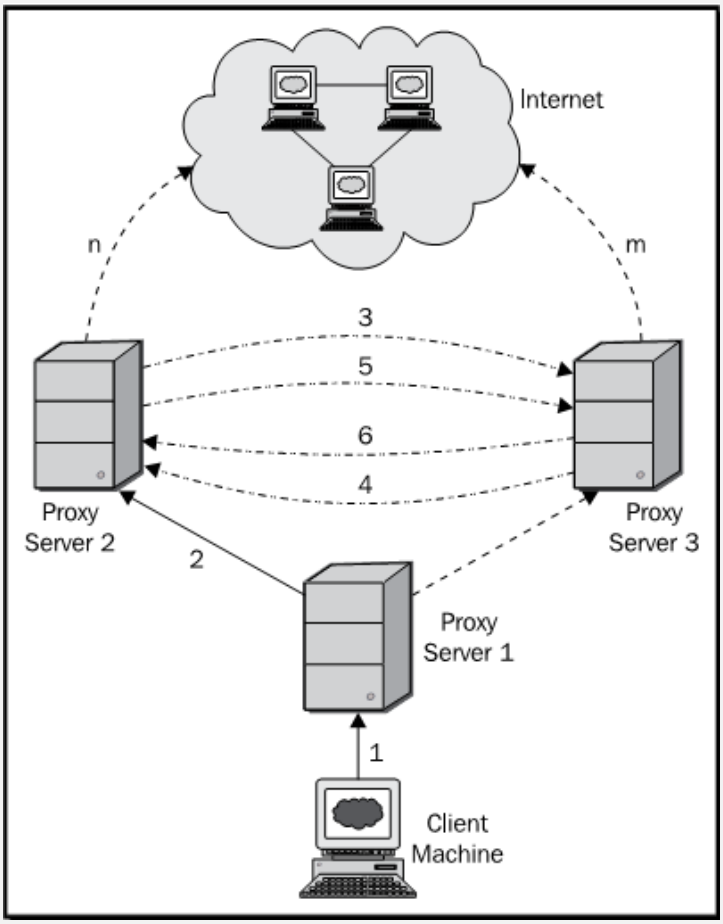
دلایل استفاده از ارتباطات گروهی کش سرورها

دلایلی بسیار زیادی برای استفاده از این نوع ارتباطات وجود دارد که به برخی از آن‌ها در بالا اشاره شد فرض کنید در یک شبکه که سطح امنیت درنظر گرفته شده در آن بسیار بالا است حال در این شبکه میخواهیم سروری که قرار است کاربران از طریق آن به اینترنت دسترسی داشته باشند پشت یک دیوار آتش قرار داشته باشد در این صورت یکی از امن ترین و بهترین راهها برای برقراری ارتباط میان کاربران و اینترنت استفاده از ارتباطات parent یا sibiling است به طوریکه کش سرورهای قبل و بعد از دیوار آتش مانند یک پل ارتباطی عمل می کنند و شبکه ای که کاربران در آن قرار می‌گیرند به طور کامل از شبکه ی خارجی ایزوله می شود شاید این سؤال پیش بیاید که راههای متعدد دیگری برای این کار وجود دارد و نیازی به استفاده از چند سرور نباشد اما فرض کنید در یک سازمان بزرگ که روزانه چند صد گیگابایت داده میان مراکز مختلف و شبکه‌های خارجی مبادله می شود و این درحالیست که همواره میزان پهنای موجود محدود بوده و از ارزش بالایی برخوردار است طبیعتاً در این حالت کش سرورهای موجود در شبکه می توانند با برقراری ارتباط با یکدیگر منابع سخت افزاری و نرم افزاری خود را با یکدیگر به اشتراک گذاشته و موجب بهبود کیفیت ودر نتیجه رضایت کاربران شوند .از جمله دلایل دیگر کاهش زمان بارگذاری صفحات هنگام بازدید کاربران است دلیل آن هم این است در صورتکیه یک درخواست در یک سرور به عنوان MISS یا کش نشده شناخته شود همان درخواست در سرور دیگر ممکن است از قبل وجود داشته باشد و به سرعت به کاربر درخواست کننده پاسخ دهد همچنین به کارگیری چند کش سرور به همراه وب سایتهایی که ترافیک روزانه آن‌ها بسیار زیاد است موجب بهبود کارایی وب سایتها از طریق ویژگی پراکسی معکوس که یکی از نمونه‌های مهم افزایش کارایی شبکه از طریق شبکه های توزیع و تحویل محتوا است می‌باشدکه در این باره در فصل های آتی مطالب مفصلی ارائه می شود همچنین با استفاده از این سرورها می توان درخواستهای مختلف را برروی منابع مختلف تأمین کننده ی پهنای باند توزیع کرد که این کار در ساعاتی که ترافیک شبکه بالا است باعث افزایش چشمگیر کیفیت بازدید کاربران می شود.

معایب استفاده از ارتباطات گروهی

هنگامی که ما جزئی از ارتباطات سلسله مراتبی میان کش سرورها باشیم تقریباً تمامی آنچه که توسط سرورهای همسایه ذخیره می شود به طور مستقیم در اختیار کاربران شبکه ما نیز قرار می‌گیرد.یکی از مهمترین نگرانی‌هایی که در این زمینه وجود دارد این است ممکن است در اثر مشکلات ناشی از پیکربندی پراکسی سرور بخش‌های مختلف سازمان و یا شبکه ی بزرگی که ما در آن قرار گرفته ایم اطلاعات به روز و معتبر در اختیار کاربران قرار نگیرد که این باعث ایجاد مشکلات عدیده ای خواهد شد همچنین در سناریوهای این چنینی آلوده شدن اطلاعات یک پراکسی سرور به انواع ویروس ها و بدافزارها ممکن است باعث آلوده شدن سایر بخشهای شبکه از طریق تبادل اطلاعات میان سرورهای مختلف شود که این

مسأله ممکن است موجب آلوده شدن تمامی کاربران و بروز یک مشکل امنیتی بزرگ شود به همین دلیل همواره باید از کیفیت پیکربندی سایر سرورها و میزان امنیت آن‌ها آگاه بوده و امنیت و کیفیت اطلاعات را فدای افزایش کارایی سرور های خود نکنیم.یکی دیگر از مسائل مهمی که در این زمینه مطرح است افشای اطلاعات محرمانه و شخصی کاربران و انتقال آن به بخش‌های مختلف شبکه ودسترسی احتمالی سایرین به این اطلاعات است که درز اطلاعات از بخش‌های مهم یک سازمان ممکن است بسیار خطرناک باشد.اشکال دیگری که در این ارتباطات مطرح است ایجاد حلقه میان سرورها و عدم دسترسی کاربران به اطلاعات از طریق منبع اصلی آن در صورت لزوم است تصویر زیر این مشکل را به تصویر کشیده است:



تصویر ۸-۲ چگونگی ایجاد حلقه را به تصویر می کشد.

در این تصویر یک درخواست از طرف کاربر به سوی پراکی سرور ۱ فرستاده می شود درصورتکیه نتواند به درخواست کاربر پاسخ دهد درخواست برای پاسخگویی به پراکسی سرور شماره ۲ فرستاده می شود به این ترتیب در صورتکیه پاسخ درخواست در پراکسی سرور های بعدی وجود نداشته باشد درخواست کاربر میان تمامی پراکسی سرور ها به طور مداوم و به صورت حلقه گردش می کند وکاربر پاسخ درخواست خود را دریافت نمی کند این خود مشکل بزرگی است که موجب نارضایتی کاربران می شود.دلیل بروز این مشکل وجود نقص در تنظیمات پیکربندی اسکوئید است و با انجام تنظیمات مناسب برروی پراکسی

سرور ها می توان از بروز این گونه مشکلات جلوگیری به عمل آورد.یک راه سریع و آسان برای حل این مشکل تغییر مقدار دستور via به on است در این حالت اسکوئید هدر via را به درخواستهای موجوددر صورت نیاز اضافه می کند وآن را به اطلاعات سایر پراکسی سرورهای شبکه نیز می رساند در صورت تکرار این درخواست ،سرورهای همسایه به جای تلاش برای پاسخ گویی به این درخواست خاص که حامل هدر via است یک پیغام خطا را به جای تلاش برای پاسخگویی و هدر دادن منابع سرور برای کاربر نمایش می دهد.راه حل دیگر برای حل این مسئله تغییر در تنظیمات ارتباطات میان سرورهاست به مثال زیر توجه کنید:

فرض کنیم دو سرور با آدرس های (192.0.2.25) s1.example.com و (198.51.100.86) s2.example.com

وجود دارد که تنظیمات ارتباطی این دو به صورت زیر است:

```
cache_peer s2.example.com sibling 3128 3130
cache_peer s1.example.com sibling 3128 3130
```

تنظیمات این چنینی برروی این دوسرور باعث ایجاد حلقه می شود که برای جلوگیری از بروز چنین مشکلاتی تنظیمات آنها را به صورت زیر اصلاح می کنیم:

تنظیمات اصلاح شده برای سرور s1.example.com به صورت زیر است:

```
cache_peer s2.example.com sibling 3128 3130
acl sibling2 src 198.51.100.86
cache_peer_access s2.example.com deny sibling2
```

به طور مشابه برای سرور s2.example.com باید به صورت:

```
cache_peer s1.example.com sibling 3128 3130
acl sibling1 src 192.0.2.25
cache_peer_access s2.example.com deny sibling1
```

اصلاح تنظیمات پیکربندی صورتی که گفته شد از ایجاد حلقه در پاسخگویی درخواستها جلوگیری می کند.

اتصال به سایرکش سرورها

در فصلهای گذشته درمورد در دستور cache_peer مطالبی مطرح شد در این فصل به بررسی جزئیات این دستور و گزینه های موجود در آن می پردازیم. ساختار کلی استفاده از دستور cache_peer به صورت زیر است:

```
cache_peer HOSTNAME TYPE HTTP_PORT ICP_OR_HTCP_PORT [OPTIONS]
```

پارامتر HOSTNAME آدرس ip و یا دامنه پراکسی سروری که قرار است با آن ارتباط داشته باشیم را مشخص می کند .پارامتر TYPE یکی از انواع گزینه های parent،multicast ویا sibling را شامل می شود که در واقع نوع برقراری ارتباط با سرورمورد نظر را تعیین می کند.

نکته: توجه داشته باشید در صورتکیه از آدرس دامنه به جای **ip** *استفاده می کنید حتما یک یا چندسرور* **dns** *را برای ترجمه ی آدرس دامنه به* **ip** *سرور مورد نظر به فایل پیکربندی اسکوئید یا سرور اضافه کنید.*

HTTP_PORT ،شماره ی درگاهی را که پراکسی سرور همسایه از طریق آن درخواستهای http را می پذیرد مشخص می کند که مقدار پیش فرض آن۳۱۲۸ می باشد.پارامتر ICP_OR_HTCP_PORT شماره ی درگاههای ICP یا HTCP جهت برقراری ارتباط با سرور سرورهمسایه را مشخص می کند که مقدار پیش فرض ICP ۳۱۳۰ که توصیه می شود این مقدار را با مقادیر دیگر جایگزین کنیدهمچنین در صورتکیه بخواهید از پروتکل HTCP نیز سرورها استفاده کنید باید مانند ICP_port ،htcp_port را نیز به همراه شماره درگاه آن به دستور بالا اضافه کنید همچنین مقدار پیش فرض درگاه htcp ،۴۸۲۷ تعیین شده است .در صورتکیه مایل به برقراری ارتباط از طریق ICP یا HTCP نباشیم کافی است شماره ی درگاهها را به صفر تغییر دهیم.از طریق خطوط زیر دو پراکسی سرور همسایه را به اسکوئید معرفی می کنیم:

```
cache_peer parent.example.com parent 3128 3130 default
cache_peer sib.example.com sibling 3128 3130 proxy-only
```

بنابراین طبق تنظیمات بالا parent.example.com یک پراکسی سرور از نوع ارتباط parent و sib.example.com یک پراکسی سرور از نوع ارتباط sibling می باشد.در ادامه به برسی جزئیات گزینه های موجود در تنظیمات پراکسی رورهای همسایه می پردازیم.

تنظیمات ICP

زمانی که بخواهیم از طریق پروتکل ICP با کش سرورهای همسایه های خود ارتباط داشته باشیم باید از پیکربندی صحیح دستورات icp_port و icp_access آگاهی داشته باشیم.در ادامه به بررسی گزینه های همراه با ICP در cache_peer می پردازیم:

no-query

در صورت استفاده از دستور no-query ،اسکوئید هیچ گاه پیام های ICP را جهت برقراری ارتباط به سوی سرور همسایه ارسال نمی کند.

multicast-responder

این دستور مشخص می کند که این سرور همسایه عضوگروه multicast بوده و اسکوئید نباید درخواستهای ارتباط ICP را به صورت مستقیم به سوی این سرور ارسال کند در حالیکه سرور پراکسی موجود همچنان قادر به دریافت پاسخ ها از این میزبان خواهد بود.

closest-only

زمانی که دستور closest-only به کار رود و پاسخ های ICP_OP_MISS وجود داشته باشد اسکوئید قادر به ارجاع درخواستهایی از نوع FIRST_PARENT_MISS نخواهد بود اما همچنان قادر به ارجاع درخواستهای نوع CLOSEST_PARENT_MISS می باشد.

background-ping

دستور round trip time اسکوئید را مجاب به ارسال درخواستهای ICP به سوی میزبان مورد نظر می کند این عمل در حالت مخفی انجام شده و تعداد تکرار آن بسیار کم است و مهمترین دلیل این کار به نوعی بررسی مسیر دریافت و ارسال داده به میزبان مورد نظر است.

تنظیمات HTCP

هنگامی که بخواهیم در ارتباطات با پراکسی سرورهای همسایه از پروتکل HTCP استفاده کنیم ابتدا باید از تنظیمات صحیح htcp_port و htcp_access در فایل پیکربندی اسکوئید اطمینان داشته باشیم.

HTCP

اگر بخواهیم به جای ICP از HTCP در ارتباطات خود استفاده کنیم،باید پارامتر htcp را به پارامتر ICP_PORT الحاق کنیم همچنین باید به جای شماره ی پورت ۳۱۳۰ ،شماره ی ۴۸۲۷ را که شماره ی پورت HTCP است قرار دهیم.فهرست دستورهایی که با htcp قابل استفاده است در زیر به همراه توضیحات آمده است:

htcp=oldsquid

در صورت استفاده از دستور htcp=oldsquid ،اسکوئید با همسایه ی خود مانند یک اسکوئید نسخه ی ۲.۵ و قبل از آن رفتار می کند و درخواستهای htcp را به سوی آن ارسال می کند.

htcp=no-clr

با استفاده از دستور htcp=no-clr اسکوئید پیام های HTCP را به سوی همسایه ی خود ارسال می کند اما پیام های CLR را ارسال نمی کند.به کارگیری این دستور همراه با htcp=only-clr باعث ایجاد ناسازگاری می شود و نباید با هم در یک فایل پیکربندی مورد استفاده قرار گیرند.

htcp=only-clr

این دستور اسکوئید را مجاب به ارسال درخواست های HTCP CLR به سوی همسایه خود می کند.

htcp=no-purge-clr

دستور htcp=no-purge-clr اسکوئید را موظف می کند تا پیام های HTCP را تنها زمانی که تنیجه ای درخواستهای purge دریافت نکند ارسال می کند.

htcp=forward-clr

این دستور درخواستهای htcp clr را دریافت کرده و آن‌ها را به سوی سرورهای همسایه ارسال می کند.

انتخاب میان سرورهای همسایه و همکار

در صورتکیه در فایل پیکربندی اسکوئید بیش از یک cache peer یا همسایه وجود داشته باشد ،این نگرانی وجود دارد که چگونه اسکوئید درخواستهای کاربران و یا پیام های htcp و icp را به سوی آن‌ها ارسال نماید دستورات زیر به منظور حل این مشکل ایجاد شده‌اند و به طور پیش‌فرض ICP برای انتخاب کش سرورهای همکار ها به کار می‌رود (درخواستهای icp را به سوی سایر پراکسی سرورهای همسایه ارسال می کند).

default

در صورتیکه همراه با تعریف پراکسی سرور همسایه از این دستور استفاده کنیم ،تنها این سرور به عنوان همسایه ی والد جهت ارتباط تعریف شده و در صورتکیه پراکسی سرورهای دیگری را هم به عنوان همسایه تعریف کنیم همه ی آن‌ها نادیده گرفته می‌شوند و اجازه ی تبادل داده به سوی سرور ما را نخواهند داشت. توجه داشته باشید در صورتکیه می خواهید بیش از یک

سرور همسایه را تعریف کنید این دستور را به کار نبرید زیرا باعث می شود فقط اولین سروری که همراه با این دستور تعریف شده به عنوان همسایه در نظر درگفته شود و سایرین نادیده گرفته شوند.

Round-robin

این توزیع ویژگی توزیع بار را میان همسایه ها و سرور شما فعال می کند.این دستور تنها زمانی مفید است که تعداد پراکسی سرورهای همسایه ی شما حداقل دو مورد باشند در غیر این صورت استفاده از این دستور کاربرد چندانی نخواهد داشت.همچنین توجه داشته باشید الگوریتم مورد استفاده در سیستم توزیع بار بسیار ساده بوده و شامل تنظیمات بخصوصی نمی باشد.

Weighted-round-robin

دستور weighted-round-robin خاصیت توزیع بار را فعال کرده و درخواستهای کاربران را در میان سایر سرورهای همسایه تقسیم می کندانتخاب بهترین همسایه و ارسال درخواستها به آن از طریق round trip time و از طریق دستور background-ping انجام می شود درواقع اسکوئید ابتدا مسیر انتقال اطلاعات به سایر همسایه ها را از نظر کیفیت سنجیده و سپس اقدام به ارسال درخواستهای کاربران به آن‌ها می کند و معمولا سرورهای والد از اولویت بیشتری نسبت به سایرین برخوردار هستند.به علاوه با استفاده از پارامتر weight می توان میزان بار توزیع شده میان سرورهای همکار را تعیین کرد.

Userhash

دستور userhash فرایند توزیع بار را برمبنای دستورهای proxy_auth و ident و نامهای کاربری تعریف شده در آن‌ها میان سایر سرورهای همسایه تقسیم می کند درواقع در اینجا هر نام کاربری تعریف شده به عنوان یک کاربر مجزا در نظر گرفته شده تعداد کاربران موجود میان سرورها تقسیم می شود.

Sourcehash

این دستور عملکردی مشابه userhash دارد با این تفاوت که به جای نامهای کاربری از آدرس ip کاربران به منظور عمل توزیع بار و یا به عبارتی توزیع درخواستهای کاربران میان سایر کش سرور ها استفاده می کند.

Carp

Cache Array Routing Protocol (CARP) نیز به منظور انجام توزیع بار میان چند سرور پراکسی به کار می‌رود. این نوع پروتکل ارتباطی درخواستهای http را میان سرورهای مختلف به صورت منظم پخش می کند .جهت کسب اطلاعات بیشتر در این زمینه به آدرس http://icp.ircache.net/carp.txt مراجعه کنید.ویژگی برجسته ی این دستور تأکید زیاد آن بر پخش یکسان درخواستها میان تمامی سرورهای پراکسی همسایه و والد در حالت عادی است در حالیکه به کارگیری دستور

weight می توان توازن میان سرورهای همسایه را بر هم زد و سهم برخی از آن‌ها بیش از سایرین در پاسخگویی در خواستها باشد .

Multicast-siblings

دستور multicast-siblings فقط زمانی به کار می رود که پراکسی سرور های موجود عضو گروهی مشخص از شبکه باشند همچنین پراکسی سرورهای موجود باید از طریق راهکارارتباطیsibling با یکدیگر در ارتباط باشند این دستور برای مواردی که میخواهیم از گروهی از پراکسی سرورهای موجود در یک گروه به عنوان پشتیبان (redundant) استفاده کنیم که معمولا با عنوان cluster sibling شناخته می شوند و عمدتا به منظور سرعت بخشیدن به شبکه و جابجایی آسان پیامهای ICP میان پراکسی سرورهای موجود به کار می رود.

تنظیمات مربوط به روش های انتخاب سرورهای همسایه و همکار

در ادامه ی معرفی گزینه های مورد استفاده در cache peer به معرفی گروهی از دستورهایی می پردازیم مکه عمدتا به منظور بهبود کارایی و کیفیتتوزیع بار میان پراکسی سرورهای شبکه به کار می روند.

weight

دستور(weight=N) weight به منظور برهم زدن میزان توازن بار میان پراکسی سرورهای موجود به کار می رود به بیان ساده تر weight میزان دریافت و ارسال داده ها در پراکسی سرورها را مشخص می کند و هرچه مقدار weight برای یک پراکسی سرور بیش از سایرین باشد میزان داده های که به سوی آن ارسال می شود بیش از سایرین است .در صورتکیه با مفهوم توزیع بار و iproute در لینوکس آشنایی داشته باشید به خوبی عملکرد این دستور را درک خواهید کرد.به طور پیش فرض مقدار تعیین شده برای دستور weight یک تعیین شده است که به معنای توازن کاملا یکسان بار در میان تمامی سرورهای پراکسی است.

basetime

(basetime=N) basetime مدت زمانی که از زمان رفت و برگشت درخواست ICP میان پراکسی سرور ها به طور یکسان کسر می شود قبل از اینکه مسیر انتخابی برای پاسخ به درخواست مورد نظر از سوی پراکسی سرورها تعیین شود. در واقع این مقدار به طور یکسان قبل از اینکه یکی از پراکسی سرور ها که زمان پاسخ دهی کمتری دارد به عنوان (پاسخگو)responser انتخاب شود تعیین می شود.

ttl

(ttl=N) ttl برای مشخص کردن زمان رفت و برگشت یک بسته ی داده در پراکسی سرورهای عضو یک گروه به کار می رود Time to Live (TTL) زمان رفت و برگشت پیام های ICP را مشخص می کند و سایر سرورهای پراکسی برای اینکه بتوانند پیام های ICP را به درستی دریافت کنند باید همراه با دستور multicast-responder پیکربندی شده باشند.

no-delay

در صورتکیه در پراکسی سرورهای همسایه و همکاری که اسکوئید با آنها در ارتباط است از delay pool به منظور کنترل سرعت دریافت و ارسال دادها توسط کاربران استفاده می کنند درخواستهای ارسال شده سایر کاربران موجود در شبکه نیز تحت تاثیر این محدودیت سرعت قرار می گیرند و باعث ایجاد تاخیر در پاسخگویی از طرف پراکسی سرورهای همسایه و در نتیجه کاهش بازده شبکه می شود برای جلوگیری از این مشکل و اینکه کاربران سایر شبکه ها و پراکسی سرورها تحت تاثیر محدودیت های موجود در delay pool ها قرار نگیرند از دستور no-delay استفاده می کنیم بدین ترتیب درخواستهای ارسال شده از طرف سایر پراکسی سرورها با نهایت سرعت ممکن پاسخ داده می شوند.

ssl

در صورتکیه این گزینه استفاده شود ارتباطات میان پراکسی سرور موجود از طریق ssl وtls رمز نگاری می شود.

Sslcert

sslcert (sslcert=FILE) مسیر دقیق فایلی که گواهینامه ی ssl را که در ارتباط با کش سرور مورد استفاده قرار می گیرد را مشخص می کند.

Sslkey

sslkey (sslkey=FILE) یکی از دستورات دلخواه است که مسیر دقیق نگهداری فایل حاوی کلید ssl را مشخص می کند و در صورت استفاده از دستور sslcert فایل حاوی کلید شخصی ssl با این دستور به صورت ترکیبی عمل می کنند.

Sslversion

فرمان sslversion (sslversion=NUMBER) نسخه ی پروتکل ssl /tls مورد استفاده را مشخص می کند به طور کلی گزینه های درنظرگرفته شده برای دستور sslversion به صورت زیر است:

۱. Automatic detection یا شناسایی خودکار: مقدار پیش فرض تعیین شده است و شماره ی نسخه را به صورت خودکار شناسایی می کند.

۲. SSLv2 only

۳. SSLv3 only

۴. TLSv1 only

۵.

Sslcipher

sslcipher (sslcipher=COLON_SEPARATED_LIST) به منظور استفاده از پروتکل openssl ایجاد شده است جهت کسب اطلاعات بیشتر در مورد رمزهای پشتیبانی شده توسط openssl به آدرس http://www.openssl.org/docs/apps/ciphers.html مراجعه فرمائید.توجه داشته باشید که رمزهای مورداستفاده می بایست موردپشتیبانی نسخه ی مورد استفاده ی openssl قرار گرفته باشد.

front-end-https

این دستور یکی از هدر های http به نام Front-End-Https: On را فعال می کند که در نرم افزار outlook شرکت مایکروسافت مورد استفاده قرار می گیرد. جهت کسب اطلاعات کاملی در مورد دلایل استفاده از این هدر به آدرس http://support.microsoft.com/kb/307347 مراجعه کنید.

سایر تنظیمات مربوط به کش سرورهای همکار

login=username:password برخی از پراکسی سرورهایی که به عنوان همسایه انتخاب شده اند از سیستم احراز هویت استفاده می کنند و جهت برقراری ارتباط نیازمند نام کاربری و رمز عبور هستند .دستور (login=username:password) نام کاربری و رمز عبور موردنیاز را به سایر تنظیمات مربوط به پراکسی سرور همسایه و یا همکار اضافه می کند.

login=PASS

دستور login=PASS در مواقعی که می خواهیم جزئیات ورود کاربران به پراکسی سرور را در اختیار یکی از همسایه های خود قرار دهیم به کار می رود در حالیکه سیستم احراز هویت اسکوئید شرایط لازم برای این کار را در اختیار ندارد به علاوه در صورتکیه اسکوئید هیچ گونه پیام احراز هویتی را از طرف کاربران دریافت نکند ،اما نامهای کاربری و رمز عبور از طریق acl های خارجی =user و =password در دسترس باشد ممکن است به جای پیام های احراز هویت ارسال شود. درصورتکیه بخواهیم برروی پراکسی سرورخود مانند سرور پراکسی همسایه سیاست احراز هویت را پیاده سازی کنیم باید هر دو سرور پایگاه داده اطلاعات کاربران خود را میان همدیگر به اشتراک بگذارند دلیل این کار همسان سازی اطلاعات کاربران به منظور حفظ یکپارچگی وجلوگیری از مشکلات ناشی از عدم شناخت کاربران است.

login=PASSTHRU

login=PASSTHRU به منظور ارجاع هدر های http که حاوی پیام‌های احراز هویت(Proxy-Authentication و WWW-Authorization) هستند به سوی پراکسی سرورهای همسایه به کار می‌رود. در این نوع برخلاف موارد قبلی که نیازمند نام کاربری و رمز عبور بود در این دستور بدون هیچ گونه احراز هویتی این پیام‌ها برای سایر سرورهای پراکسی همسایه ارسال می شود.

login=NEGOTIATE

دستور login=NEGOTIATE زمانی مفید است که سرور پراکسی شما عضو یک گروه از سرورها باشد و سایر سرورهای همسایه به منظور برقراری ارتباط نیازمند احراز هویت پراکسی سرور شما هستند در اینجا Service Principal Name (SPN) توسط مقدار KRB5_KTNAME مورد استفاده قرار می گیرد.

Connect-timeout

connect-timeout ،میزان زمان تأخیر را برای سرور همسایه را تعیین می کند که این مقدار برای سرورهای مختلف می تواند متفاوت باشد.درصورتکیه این مقدار تعریف نشده باشد ،اسکوئید این مقدار را براساس دستور peer_connect_timeout

تعریف می کند.

Connect-fail-limit

connect-fail-limit (connect-fail-limit=N) تعداد اتصالات ناموفق با یک پراکسی سرور همسایه را تعیین می کند پس از اتمام این مقادیر ،سرور مذکور به عنوان خارج از سرویس تلقی شده و هیچ اتصال دیگری به سوی آن برقرار نمی شود.مقدار پیش‌فرض connect-fail-limit ، ۱۰ تعیین شده است.

Max-conn

تعداد اتصالاتی که به طور همزمان و موازی ای که می توان با یک سرور پراکسی همسایه برقرار کرد به توسط دستور max-conn (max-conn=N) تعیین می شود.

Name

در برخی موارد ممکن است تعدادی از پراکسی سرورهای یک گروه دارای نام میزبان یکسانی باشند سرور پراکسی از یک درگاه برای گوش کردن به اتصالات برقرار شده با آنان استفاده کند در این مواقع یک نام میزبان به تنهایی قادر به ایجاد تمایز میان سرورها ی مختلف نیست به همبن دلیل با استفاده از دستور name (name=STRING) می توان یک نام منحصر به فرد به هر یک از ماشین‌ها اختصاص داد . به طور پیش‌فرض این دستور از نام های میزبان ،آدرس ip یا آدرس دامنه سرورهای

همسایه به عنوان این نام منحصر به فرد استفاده می کند.همچنین نام انتخاب شده در این دستور می تواند در دستورهای دیگری نظیر cache_peer_access و cache_peer_domain مورد استفاده قرار گیرد و به عبارتی این نام منحصر به فرد به عنوان یک مقدار معتبر می تواند در این دستورها دارای اعتبار کافی باشد.

proxy-only

به طور معمول اسکوئید تمامی درخواستهایی که از طرف سرورهای همسایه که به منظور پاسخگویی به درخواستهای کاربران ارسال شده است را در صورتی که قابلیت نگهداری داشته باشند در دیسک سخت خود نگهداری می کند از سوی دیگر در صورتی پراکسی سرورهای همسایه قادر به پاسخگویی به درخواستهای کاربران باشند نگهداری دوباره همان اطلاعات در دیسک سخت می تواند موجب هدر رفتن فضای دیسک سخت شود.در صورتکیه پراکسی سرورهای همکار بتوانند با سرعت بالا و قابل قبول وبدون ایجاد هیچ گونه تأخیری به درخواستهای کاربران شبکه شما پاسخ دهند با استفاده از دستور proxy-only می توان اسکوئید را مجاب کرد که هیچ گونه پاسخ دریافت کرده ازسرور همسایه را در حافظه ی خود نگهداری نکند.توجه داشته باشید تنها زمانی از این دستور استفاده کنید که سرور همسایه شما بدون کوچکترین تأخیر و بالاترین سرعت ممکن بتواند به درخواستهای کاربران شبکه شما پاسخ دهد در غیر این صورت به کارگیری این دستور موجب ایجاد تاخیرهای پی درپی در شبکه وکاهش کیفیت خواهد شد.

Allow-miss

درخواستهای کاربران زمانی به سوی پراکسی سرورهای همسایه‌ای که از ارتباط sibling با پراکسی سرور شما استفاده می کنند ارسال می شود که این درخواستها از نوع hit یا از قبل کش شده باشد.allow-miss باعث ارسال درخواستهای از نوع miss یا کش نشده به سوی پراکسی سرورهای از نوع sibling می شود در استفاده از این دستور دقت داشته باشید زیرا ممکن است باعث ایجاد حلقه ها و مشکلات از این دست شود.

کنترل ارتباطات با کش سرورهای همکار

تاکنون پارامترهای گوناگونی در رابطه با چگونگی برقراری ارتباط با سرورهای همسایه آموختیم در ادامه می‌خواهیم درمورد کنترل دسترسی به سرورهای همسایه و همچنین کنترل ارجاع درخواستها و اطلاعات مختلف به سایر سرورها مطالبی با جزئیات بیشتری را بیاموزیم.

ارجاع از طریق آدرسهای دامنه کش سرورها

دستور cache_peer_domain به منظور تعیین آدرس دامنه سرورهای همسایه به کار می‌رود درواقع با استفاده از این دستور می توان آدرس های دامنه مربوط به کش سرورهایی که می خواهیم با آنها از طریق نشانی دامنه ارتباط داشته باشیم را مشخص کنیم .ساختار کلی این دستور به صورت زیر است:

cache_peer_domain NAME [!]domain [[!]domain] ...

دراینجا عبارت NAME نام کش سرور همسایه را مشخص می کند و در قسمت دومین نام دامنه کش سرور مورد نظر قرار می‌گیرد .توجه داشته باشید با استفاده از این دستور می توان به تعداد دلخواه نام های دامنه را تعریف کرده و به توسط یک فاصله آن‌ها را از هم جدا کرد همچنین علامت ! آدرس subdomain هایی که مایل به اضافه کردن آن‌ها در لیست نامهای دامنه مجاز نیستیم و تمایلی به برقراری ارتباط با آن‌ها نداریم را مشخص می کند.مثال زیرخود به خوبی گویای این مسأله است:

```
cache_peer   parent.example.com   parent  3128   3130   default proxy-only
cache_peer   acad.example.com   parent  3128   3130   proxy-only
cache_peer   video.example.com  parent  3128   3130   proxy-only
cache_peer   social.example.com   parent  3128   3130   proxy-only
cache_peer  search.example.com  parent  3128   3130  proxy-only
cache_peer_domain   acad.example.com   .edu
cache_peer_domain   video.example.com   .youtube.com .vimeo.com
cache_peer_domain   video.example.com   .metacafe.com .dailymotion.com
cache_peer_domain   social.example.com  .facebook.com .myspace.com
cache_peer_domain   social.example.com   .twitter.com
cache_peer_domain   search.example.com   .google.com .yahoo.com
```

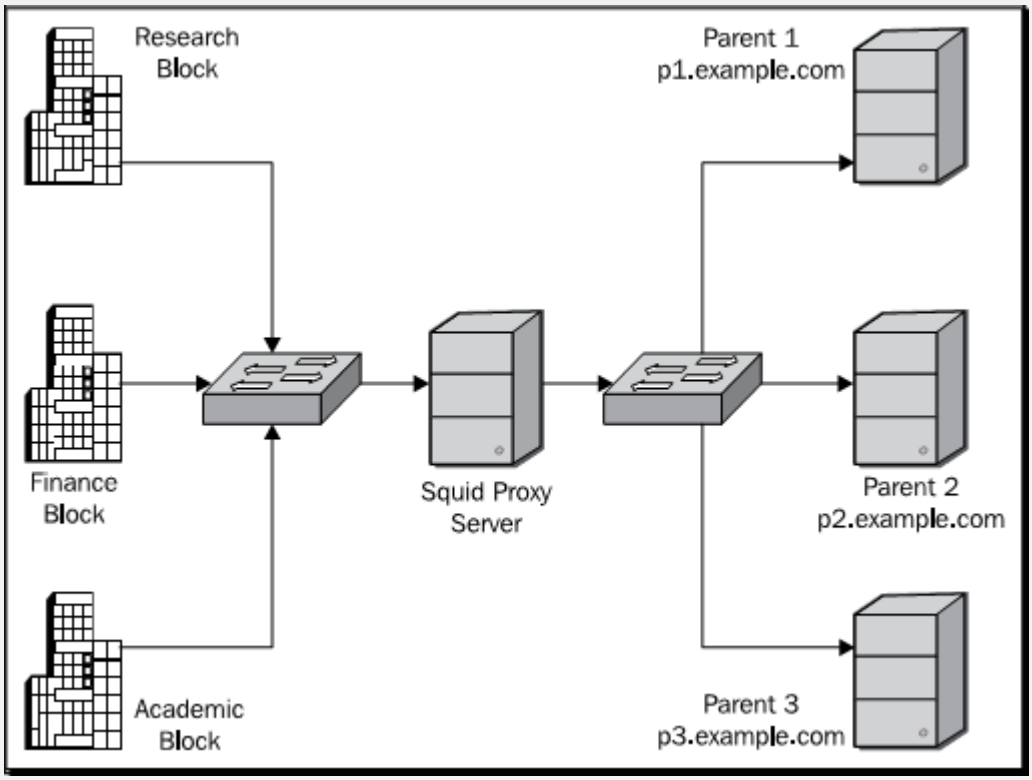
دراین مثال پراکسی سرور acad.example.com فقط قادر به ارجاع درخواستهایی است که از طرف دامنه edu. دریافت شده است در صورتکیه نام هریک از سرورهای همسایه در لیست نام دامنه سرورها قرار نگیرد می تواند تمامی درخواستهای موجود را به سوی کش سرور ما ارجاع دهد.همانطور که می‌بینید این دستور به راحتی می تواند حوزه ی فعالیت سرورهای همسایه و همچنین سطح ارتباط آن‌ها را مدیریت و کنترل کند.

Cache peer access

cache_peer_access که نمونه ی انعطاف پذیر تر دستور cache_peer_domain محسوب می شود که می تواند سطح ارتباطات و دسترسی سایر سرورهای پراکسی را با استفاده از ACL ها مدیریت کند.ساختار کلی دستور cache_peer_access به صورت زیر است:

cache_peer_access NAME allow|deny [!]ACL_NAME [[!]ACL_NAME] ...

پارامتر NAME مانند آنچه که در cache_peer_domain تعریف شد نام کش سرور همسایه را مشخص می کند همچنین پارامتر allow |deny اجازه ی و یا عدم اجازه ی ارجاع یک درخواست به این کش سرور را مشخص می کند.فرض کنیم سرور ما دارای سه پراکسی سرور همسایه به نامهای p3.example.com , p1.example.com و p3.example.com است ،پراکسی سرور p3.example.com از طریق خطوط ارتباطی سریع و قابل اطمینان به اینترنت متصل است و به همین دلیل سرور باارزشی محسوب می شود در مقابل سرورهای p1 و p2 از خطوارتباطی ارزان ولی از قابلیت اطمینان پایین‌تری برخوردار هستند همچنین در شبکه ای که این سرورها قرار دارند سه subnet به نامهای academic , research و finance وجود دارد مطابق تصویر زیر بخش‌های مختلف سازمان را تشکیل می دهند:



تصویر ۸-۳ سطح ارتباط چند کش سرور در یک سازمان را نشان می دهد.

تنظیمات پیکربندی این سه سرور به صورت زیر است:

```
cache_peer p1.example.com parent 3128 3130 round-robin
cache_peer p2.example.com parent 3128 3130 round-robin
cache_peer p3.example.com parent 8080 3130
acl academic src 192.0.2.0/16
acl finance src 198.51.100.0/16
acl research src 203.0.13.0/16
acl imp_domains dstdomain .corporate.example.com .edu
acl ftp proto FTP
cache_peer_access p3.example.com    deny ftp
cache_peer_access  p3.example.com   allow research
cache_peer_access  p3.example.com   allow academic imp_domains
 cache_peer_access   p3.example.com   allow finance imp_domains
cache_peer_access  p3.example.com   deny academic
cache_peer_access  p3.example.com   deny finance
```

دراین تنظیمات اجازه ی ارجاع درخواستهای مربوط به واحد research سازمان به سرور p3 اعطا شده است همچنین به برخی از آدرسهای دامنه خاص در واحد finance اجازه ی ارجاع درخواستها به p3 داده شده است و ارجاع تمامی درخواستهای دیگر مانندپروتکل ftp و سایر درخواستها به سرور p3 محدود شده است و فقط در صورتیکه سرورها p1 و p2 به هر دلیلی از دسترسی خارج شده باشند درخواستهای سایر بخش‌ها نیز به سوی p3 ارجاع داده خواهند شد.هدف از این کار استفاده از یک سرور مطمئن با پهنای باندمناسب و پایداری بالا به عنوان پشتیبان و استفاده از دو سرور دیگر با خصوصیات معمول به منظور برقراری ارتباطات روزانه در سازمان می‌باشد همانطور که مشاهده کردیم استفاده از ACL ها باعث انعطاف پذیری بسیار زیاد و مدیریت منسجم منابع و امکانات خواهد شد.

جابجایی در ارتباطات با سرورهای همکار

همانطوریکه قبلاً دیدیم ، می توان با استفاده از دستورات مختلفی که معرفی شدند کش سرورهای همسایه را جهت برقراری ارتباط وتبادل داده‌ها به فایل تنظیمات اسکوئید اضافه کرد این درحالیست که سرورهای همسایه داده‌های با برچسب miss یا

کش نشده را فقط برای گروهی مشخص از دامنه ها پذیرفته در حالیکه تمامی داده‌های با برچسب hit یا ذخیره شده را جهت نگهداری در دیسک سخت ،می پذیرند این ارتباطات و تبادل داده‌ها و همچنین آگاهی از وضعیت کش سرورهای همسایه از طریق پروتکل های icp , htcp و یا digest امکان‌پذیر است .هنگامی که یک داده با برچسب miss از طرف سرور همسایه به سوی سرور شما فرستاده می شود به این معناست که داده و یا به عبارت بهتر شیء مورد نظر در مخزن کش سرور همسایه وجود ندارد و او آن را از شما درخواست می کند .جابجایی در ارتباطات با سرورهای همسایه از طریق دستور neighbor_type_domain امکان‌پذیر است. ساختار کلی این دستور به صورت زیر است:

```
neighbor_type_domain CACHE_HOST parent|sibling domain [domain] ...
```

فرض کنیم sibling.example.com یک پراکسی سرور همکار است اما این سرور فقط اجازه ی ارجاع درخواستهایی که شامل دامنه edu. هستند را برای ما با و توجه به تنظیمات زیر صادر کرده است:

```
cache_peer parent.example.com parent 3128 3130 default proxy-only
cache_peer sibling.example.com sibling 3128 3130 proxy-only
neighbor_type_domain sibling.example.com parent .edu
```

کنترل بر تغییر نشانی درخواستها

هم‌اکنون با توجه به دستوراتی که تا این لحظه معرفی شده‌اند می توان پراکسی سرورها را مجاب کرد فقط به گروهی خاص از درخواستها و آدرسها پاسخ دهند و یا بالعکس گروهی دیگر از دستورات که کاربردی مشابه این دارند می توانند با مجاب کردن اسکوئید گروهی از آدرسهای خاص را که تمایلی به کش کردن آن‌ها و را ردو بدل شدن اطلاعات موجود در آن‌ها در میان سرورهای همسایه نداریم به طور مستقیم از طریق سرورهای اصلی پاسخ دهی شوند که در ادامه به معرفی آن‌ها می پردازیم.

hierarchy_stoplist

هدف اصلی ما از تبادل اطلاعات با سرورهای همسایه افزایش میزان کارایی و ذخیره داده های بیشتر در مخزن کش سرور است اما برخی از وب سایتهایی که مدام در حال بروزرسانی هستند و محتوای آن‌ها به طور پیوسته در طول روز بارها عوض می‌شوند در صورتکیه اطلاعات آن‌ها در میان سایر سرورها جابجا شود ممکن است باعث ایجاد مشکلاتی در مشاهده محتوای آن‌ها شود.دستور hierarchy_stoplist باعث می شود اسکوئید محتوای این وب سایتها را مستقیماً از سرورهای اصلی آن‌ها دریافت و بروز کرده و از ارجاع آن‌ها به سایر سرورهای همسایه جلوگیری کند.به طور کلی مکانیسم کاری این دستور به

گونه ای است که اگر کلمات خاصی که در این دستور تعریف می‌شوند در هر وب سایتی مشاهده شود از ارجاع این محتوا به سرورهای همسایه جلوگیری می کند:

```
hierarchy_stoplist cgi-bin ? Jsp
```

در صورتیکه در هر وب سایتی اطلاعاتی با این فرمت مشاهده شوند از ارجاع آن‌ها جلوگیری به عمل آمده وبروزسانی اطلاعات آن‌ها فقط از طریق سرورهای اصلی انجام می شود.

نکته:در صورتکیه دستور **never_direct** *را به همراه* **hierarchy_stoplist** *به کار ببرید ،دستور* **never_direct** *موجب توقف فعالیت دستور* **hierarchy_stoplist** *می شود.*

always_direct

این دستور عملکردی مشابه دستور hierarchy_stoplist دارد بااین تفاوت که با استفاده از always_direct می توان مستقیماً آدرس وب سایتهایی را که مایل به ارجاع آن‌ها به سرورهای دیگر نیستیم را قرار داده و همواره اطلاعات این وب سایتها فقط از طریق سرورهای اصلی و به طور مستقیم و بدون دخالت سرورهای همسایه دریافت وبروزرسانی می شود.به مثال زیر توجه کنید:

```
acl local_domain dstdomain .local.example.com
always_direct allow local_domain
```

محتوای موجود در آدرس local.example.com همواره از طریق سرورهای اصلی آن و بدون ارجاع به سایرین دریافت می شود.

never_direct

دستور never_direct درخواستهایی را که نباید مستقیا به سرورهای اصلی خود اسجاع داده شوند و می بایست ابتدا به سرورهای همسایه ی موجودارجاع داده شوند را مشخص می کند این عمل زمانی مفید است که در شبکه تمامی دادههای ارسالی به سرورهای خارجی ابتدا می بایست از یک پراکسی فایروال عبور کنند.فرض کنیم سروری با نام firewall.example.com وجود دارد که تمامی درخواستهای موجود ابتدا به آن ارجاع داده می‌شوند اما کماکان قادر به ارجاع درخواستها به سرورهای داخلی هستیم.خطوط زیر تنظیمات مربوط به پیکربندی never_direct را نشان می دهد:

```
cache_peer firewall.example.com parent 3128 3130 default
```

```
acl local_domain dstdomain .local.example.com
always_direct allow local_domain
never_direct allow all
```

با توجه به این تنظیمات تمامی درخواستهای موجود برروی سرورهای داخلی شبکه مستقیماً به سرورهای اصلی و خارجی ارجاع داده می‌شوند و تمامی داده‌هایی که از خارج از شبکه ی داخلی وارد می‌شوند ابتدا از firewall.example.com عبور کرده و سپس در اختیار کاربران مختلف قرار می گیرد.

prefer_direct

هر در خواستی که توسط اسکوئید قابل کش شدن باشد از طریق سایر کش سرورهای همسایه در صورت نیاز به سوی یکدیگر ارسال می شود دلیل این امر هم واضح است،افزایش کیفیت مشاهده صفحات توسط کاربران شبکه اما همانطور که قبلا هم اشاره شد در برخی از حالت های خاصی که پیش می آید مایل به مبادله شدن درخواستهای کاربران میان کش سرورهای مختلف نیستیم و میخواهیم یک درخواست مشخص به طور مستقیم از طریق وب سرورهای اصلی آن پاسخ داده شوند دستور prefer_direct به همین منظور ایجاد شده است که مقدار پیش فرض آن offاست به این معنی که اسکوئید همواره برای هر درخواستی ابتدا کش سرورهای همسایه را بررسی کرده و در صورتیکه آنها بتوانندرخواست را پاسخ دهی کنند به سرورهای اصلی مراجعه نخواهد کرد و لی در صورتکیه مقدار این دستور برابر on قرار داده شود ،اسکوئید ابتدا سرورهای اصلی را به جای سرورهیا همسایه به منظور پاسخ گویی به درخواست جستجو می کند.

nonhierarchical_direct

Non-hierarchical از جمله درخواستهایی هستند که توسط دستور hierarchy_stoplist قبلا در تنظیمات اسکوئید شناسایی و مشخص شده اند و از جمله محتوایی هستند که توسط اسکوئید قابلیت ذخیره سازی را ندارند این گونه درخواستها و محتوای موجود نباید به سوی پراکسی سرورهای همسایه ارجاع داده شوند زیرا هیمچ تاثیری بر میزان hit rate ندارند بهترین ایده برای رفتار با این گونه محتوا ودرخواستهای خاص ،ارجاع مستقیم آنها به سرورهای اصلی فراهم کننده ی این محتوا می باشد که این امر به وسیله دستور nonhierarchical_direct to on امکانپذیر است در صورتیکه مقدار این دستور off باشد این درخواستها ابتدا به سوی سرورهای همسایه ارسال می شود.توجه داشته باشید گرچه درخواستهای https قابل کش شدن نسیتند اما مقدار این دستور برای این درخواستها در صورتکیه از یک فایروال پراکسی عبور می کنند باید off باشد.

پروتکل های ارتباطی ارتباط با سرورهای همکار

تاکنون مطالب مفصلی در مورد چگونگی ارتباط سرورهای پراکسی اسکوئید با یکدیگر،چگونگی پیکربندی آنها ،مزایا و

معایب به اشتراک گذاری منابع و سایر خصوصیات این موضوع آموختیم.سرورهای اسکوئید به منظور برقراری ارتباط موثر با یکدیگر و تبادل اطلاعات و منابع خود از پروتکل های مختلفی همچون ICP ,HTCP و cache digest که به طور اختصاصی بدین منظور طراحی شده اند استفاده می کنند تاکنون در مورد چگونگی به کار گیری آنها مطالب زیادی آموخته ایم اما در ادامه می خواهیم با ساختار این دستورات بیشتر آشنا شویم.

Internet Cache Protocol

ICP یا Internet Cache Protocol یکی از ساده ترین پروتکل هایی است که به منظور ارتباط پراکسی سرورها با یکدیگر با هدف دریافت داده های مورد نیاز در سرورهای همسایه به کار می رود . با توجه به اینکه داده مورد نظر قبلا توسط کاربران دیگر درخواست شده باشد یا نه و یا در سرورهای همسایه دیگر وجودداشته باشد یا خیر رفتار متفاوتی را از خود نشان می دهند ،در صورتکیه در هر یک از سرورهایی که اسکوئید با آنها ارتباط دارد وجود داشته باشد در این صورت درخواست مورد نظر به آنها ارجاع داده می شود و در غیر اینصورت از سرورهای اصلی ای که محتوای مورد نظر از آن دریافت شده برای پاسخگویی استفاده می کند.یکی از مهمترین وظایف پروتکل ICP شناسایی کش سرورهای سالم و ناسالم از حیث وضعیت سرویس دهی و همچنین تشخیص سریعترین مسیر برای ارتباط با آنها است که این عمل از طریق محاسبه زمان رفت و برگشت ارسال بسته های ICP به سوی سرورهای پراکسی صورت می گیرد .به هر حال پروتکل ICP به دلیل ساختار ساده ای که دارد به راحتی قابل استفاده و پیکربندی است اما دارای مشکلات عمده ای نیز هست ،که اولین مورد تاخیر است ،زمانی که اسکوئید درخواستی را دریافت می کند نمی داند که پاسخ این درخواست در کدام یک از سرورهای پراکسی همسایه وجود دارد و یا اصلا در آن سرورها وجود دارد یا خیر برای این کار ICP تمامی سرورهای موجود را برای یافتن پاسخ درخواست جستجو می کند حال اگر تعداد پراکسی سرورهای همسایه زیاد باشد باعث ایجاد تاخیر نسبتا زیادی می شود و نتیجه ی آن نارضایتی کاربران است همچنین در صورتکیه تعداد سرورهایی که با یکدیگر در ارتباط هستند زیاد باشد در آن واحد بسته های ICP بسیار زیادی میان یکدیگر رد و بدل می کنند که در شرایط سخت کاری باعث ایجاد تصادم در شبکه خواهد شد.برای جلوگیری از ایجاد تصادم می توان از Multicast ICP استفاده کرد.از دیگر ایرادات ICP می توان به مشکلات امنیتی ،خطا در تشخیص داده های کش شده می توان اشاره کرد.برای کسب اطلاعات دقیق تر در مورد این پروتکل ارتباطی به آدرسهای http://icp.ircache.net/rfc2186.txt و http://tools.ietf.org/html/rfc2187 مراجعه کنید.با وجود مشکلاتی که در ICP وجود دارد توصیه می شود در شبکه های بزرگ باترافیک بالا از HTCP به جای ICP استفاده کنید.

Cache digests

اسکوئید فهرست تمامی داده های کش شده را حافظه ی اصلی سرور به همراه جزئیات آنها نگهداری می کند بدین ترتیب در

صورت دریافت یک درخواست جدید به سرعت می تواند حدس بزند که درخواست های ارائه شده جز داده های با برچسب hit است یا miss بدون نیاز به جستجو در میان تمامی داده های ذخیره شده برروی دیسک سخت.همچنین cache digest خلاصه ای از داده های کش شده را در یک فایل به صورت بیتی نگهداری می کند که این ساختاراصطلاحا Bloom Filter data structure نام دارد که در صفحه ی ویکی پدیای آن به آدرس http://en.wikipedia.org/wiki/Bloom_filter اطلاعات دقیق تری وجود دارد.بیت های موجود در این ساختار نشان می دهند که داده مورد نظر به صورت کش شده در دیسک سخت وجود دارد و یا خیر در صورتیکه ارزش بیت آن 0 باشد به این معنی است که داده مورد نظر کش نشده است و مقدار 1 به معنی وجود داده در دیسک سخت است.خلاصه این آمار از طریق سایر کش سرورهای همسایه نیز از طریق یک آدرس مخصوص قابل دسترسی است.در صورتکیه یک درخواست که همواره با یک آدرس url همراه است توسط کش سرور دریافت شود،با بررسی آمار cache digest که همواره بروزرسانی می شود می تواند تشخیص دهد که این درخواست در مخزن کش سرورهای موجود وجود دارد و یا خیر.یکی مزایای مهم cache digest کاهش میزان بارترافیکی ایجاد شده از طریق ارسال پیام سرورهای مجاور به یکدیگر به منظور اطلاع از وضعیت داده های درخواست شده توسط کاربران است و همان مشکلی که پروتکل ICP با آن مواجه بود یعنی ایجاد ترافیک اضافی در شبکه را حل می کند.از جمله معایب پروتکل digest می توان به مشکل تشخیص داده های با برچسب hit و miss اشاره کرد که معمولا پس از حجیم شدن فایلهای نگهداری اطلاعات داده های کش شده به وجود می آید همچنین یکی دیگر از مشکلاتی که در این رابطه وجود دارد این است که فایل نگهداری اطلاعات داده ها به طور دوره ای مثلا هر ساعت یک بار یعنی ۲۴ بار در طول یک شبانه روز بروزرسانی می شوند و داده هایی که پس از هر یک ساعت از بروزرسانی فایل کش و ذخیره می شوند تا بروزرسانی بعدی جز داده های کش نشده محسوب می شود و برچسب miss را با خود به همراه دارند که البته پس از بروزرسانی بعدی به صورت خودکار در لیست اطلاعات داده های کش شده قرار می گیرند.

پیکربندی اسکوئید و Cache Digest

به منظور استفاده از cache digest ، باید این ویژگی در زمان کامپایل اسکوئید از طریق فرمان enable-cache-digests به دستور configure/. اضافه شود.در ادامه به بررسی گزینه های مربوط به cache digest در فایل پیکربندی اسکوئید می پردازیم.

Digest generation

این دستور به منظور کنترل ایجاد فایل cache digest در اسکوئید فقط در مواقعی که اسکوئید از طریق این پروتکل با سایر سرورهای پراکسی همسایه و همکار خود به تبادل اطلاعات می پردازد استفاده می شود دستور digest_generation

دارای دو گزینه ی انتخابی on و off است ، که گزینه ی پیش فرض تعیین شده برای آن on می باشد به این معنا که اسکوئید همواره فایلهای cache digest را ساخته و بروز می کند و در صورتکیه مایل به انجام چنین کاری نیستید با وارد کردن گزینه ی off از ساخت آن جلوگیری کنید.

Digest bits per entry

همانظور که پیش تر هم اشاره شد ،اسکوئید اطلاعات مربوط به داده های کش شده را در یک فایل با استفاده از الگوریتم ذخیره سازی bllom filer و به صورت ترکیبی از بیت ها نگهداری و به نوعی آنها را به صورت کدگذاری شده ذخیره می کند و مشکلاتی همچون عدم شناسایی فایلهای کش شده نیز در این مرحله به وجود می آید فرمان digest_bits_per_entry تعداد بیتهایی را که برای کدگذاری و درج مشخصات یک شیء کش شده به کار می رود را مشخص می کند.هر چه مقدار تعیین برای این دستور بیشتر باشد میزان دقت و صحت اطلاعات ذخیره شده افزایش می یابد و بالطبع باعث مصرف بیشتر حافظه و افزایش حجم کلی فایل ذخیره سازی اطلاعات خواهد شد که میزان پهنای باند و زمان مورد نیاز برای جابجایی آن میان سرورهای همسایه را نیز افزایش می دهد. مقدار پیش فرض تعیین شده برای digest_bits_per_entry مقدار ۵ است که با رعایت میزان پهنای باند شبکه و منابع سخت افزاری سرورها می توان این مقدار را به ۷ افزایش داد که نتیجه ی آن کاهش خطاهای ناشی از ثبت اطلاعات داده های ذخیره شده می باشد.

Digest rebuild period

با استفاده از دستور digest_rebuild_period می توان زمان بروزرسانی فایل cache digest را تعیین نمود.مقدار پیش گزیده ی این دستور ۱ ساعت است به این معنی که فایل حاوی اطلاعات داده های کش شده هر یک ساعت یک بار بروزآوری می شود که این زمان نسبتا طولانی است و باتوجه به منابع سخت افزاری سرورها می توان این مقدار را کاهش داد البته به یاد داشته باشید که بروزرسانی و ذخیره اطلاعات در این فایل به دلیل ساختار ویژه بار پردازشی زیادی را به پردازنده ی سرور تحمیل می کند به همین دلیل کاهش این زمان باید با احتیاط بیشتری صورت گیرد و قبل از انجام این کار از قدرت پردازنده و میزان ترافیک موجود برروی سرور اطمینان حاصل نمائید. در صورتکیه همه چیز مهیا باشد می توانید این مقدار را از یک ساعت به هر ۱۰ یا ۱۵ دقیقه یک بار کاهش دهید که نتیجه ی این کار بروزآوری سریعتر اطلاعات داده های کش شده می باشد.

Digest rebuild chunk percentage

دستور digest_rebuild_chunk_percentage میزان داده هایی را که پس از هر بار بروزرسانی فایل cache digest به آن اضافه می شود را مشخص می کند.این میزان به صورت درصد بیان می شود و مقدار پیش فرض آن ۱۰ درصد است.

Digest swapout chunk

این دستور میزان داده هایی از فایل cache digest را که در یک زمان مشخص برروی دیسک سخت ذخیره می کند را به صورت بایت تعیین می کند که مقدار پیش فرض دستور digest_swapout_chunk، ۴۰۹۶ بایت تعیین شده است.

Digest rewrite period

میزان زمانی را که فایل cache digest به طور کامل برروی دیسک سخت ذخیره و پس از آن به توسط سایر همسایه ها قابل دریافت است را تعیین می کند.مدت زمان این فرایند از طریق دستور digest_rewrite_period تعیین می شود و معمولا برابر مدت زمانی است که cache digest بروزرسانی می شود.

آشنایی با پروتکل Hypertext Caching

HTCP یا Hypertext Caching،ساختاری تقریبا مشابه با پروتکل ICP دارد اما دارای خصوصیات ویژه ای است که در مقایسه با پروتکل ICP میزان کارایی آن به مراتب بیشتر است .پروتکل های ICP و HTCP هر دو از پروتکل UDP به منظور ارتباطات خود با سایر کش سرورها استفاده می کنند و HTCP به طور پیش فرض از پروتکل محبوب TCP به منظور رفع اشکال استفاده می کند.به طور کلی HTCP در مقایسه با ICP دارای مزایایی به شرح زیر است:

- درخواستهای ICP فقط شامل آدرس URL داده مورد نظر است در حالیکه HTCP علاوه بر آدرس کامل آن ،هدر های HTTP همراه با آن را نیز شامل می شود به باعث تمیز دقیق تر object های کش شده و جلوگیری از مشکلات ناشی از ثبت اطلاعات نادرست می شود.

- HTCP آدرس مستقیم داده کش شده را در اختیار سایر سرورهای همسایه قرار می دهد این در حالیست که ICP از این قابلیت برخوردار نیست.

- HTCP قابلیت مشاهده و بررسی فرایند حذف و اضافه ی داده ها به توسط سرورهای همسایه را فراهم آورده در حالیکه ICP این قابلیت را ندارد.

- HTCP با فراهم آوردن امکان افزایش طول پیام های دودویی از قابلیت انعطاف به مراتب بیشتری نبست به پروتکل ICP برخورداراست.

- HTCP قابلیت ها امنیتی نظیر به اشتراک گذاری کلید های تشخیص هویت را داراست در حالیکه ICP فاقد هرگونه ابزار امنیتی است .

جهت آشنایی بیشتر با پروتکل HTCP می توانید از وب صفحه ی http://tools.ietf.org/html/rfc2756 دیدن نمائید.

خلاصه فصل

تمامی مطالب این فصل به معرفی چگونگی ارتباط پراکسی سرورهای مختلف در شبکه و همچنین دلایل به اشتراک گذاری منابع وهمچنین وش های مختلف تبادل اطلاعات میان سرورهای مختلف شبکه اختصاص یافته است.به طور کلی مطالب مورد بحث در این فصل شامل موارد زیر بوده است:

- مزایا و معایب استفاده از ارتباط میان پراکسی سرورها
- بررسی دستورهای مختلف در رابطه با ارتباطات میان سرورها
- راههای افزایش امنیت در ارتباط میان سرورها
- معرفی گزینه های مختلف در مورد کنترل اطلاعات ارسال و درثافت شده از سایر سرورهای پراکسی
- معرفی و بررسی پروتکل های ارتباطی میان پراکسی سرورها به همراه بررسی مزایا و معایب هر یک.

فصل نهم

پیکربندی اسکوئید در حالت پراکسی معکوس

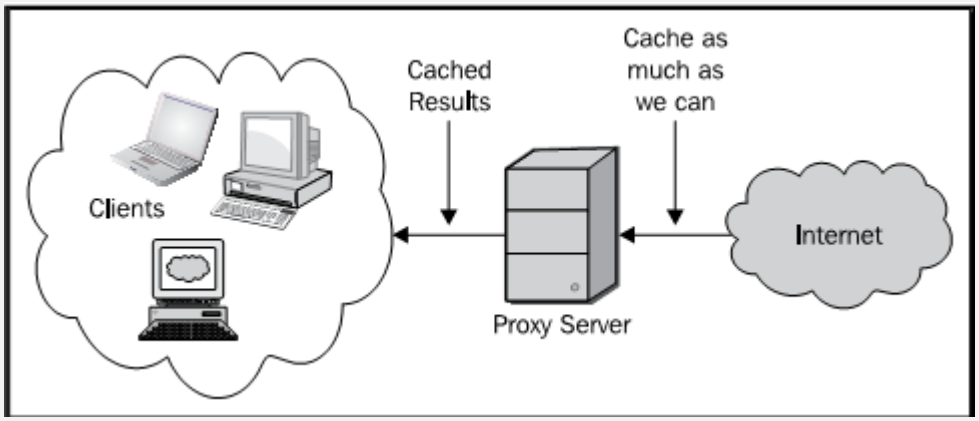
تاکنون چگونگی استفاده از اسکوئید را در موقعیت های گوناگون به منظور کش کردن درخواستهای کاربران و پنهان سازی مشخصات آنها از دید شبکه خارجی و همچنین استفاده از چند کش سرور مجزا به منظور افزایش کارایی و استفاده از حداکثری از توان پراکسی سرورها مطالبی را آموختیم.یکی دیگر از کاربردهای مهم و جالب اسکوئید ، استفاده از آن به منظور افزایش سرعت در پاسخگویی درخوساستهای کاربران به توسط وب سرورهای مقصد صورت می گیرد این روش استفاده از اسکوئید که در واقع به عنوان شتاب دهنده به سرورهای اصلی و مقصد صورت می گیرد تحت عنوان پراکسی معکوس و یا سرور جانشین شناخته می شود که در این فصل با چگونگی عملکرد آن آشنا خواهیم شد.

به طور کلی مطالب زیر در این فصل پوشش داده خواهند شد:

- آشنایی با فرایند پراکسی معکوس
- پیکربندی اسکوئید به عنوان یک سرور شتاب دهنده و یا جانشین
- آشنایی با کنترل دسترسی ها در پراکسی معکوس
- ارائه مثالهایی از نحوه پیکربندی اسکوئید در حالت پراکسی معکوس

پراکسی معکوس چیست؟

در فصل های گذشته مطالب جامعی در مورد استفاده از اسکوئید به منظور ذخیره ی داده ها ی کاربران و استفاده از آن در مراجعه های بعدی به منظور افزایش بازدهی و کیفیت وب گردی کاربران عنوان شد.این فرایند از طریق کپی برداری از داده های درخواستی کاربران و ذخیره ی آن در دیسک سخت و حافظه ی رم پراکسی سرور اسکوئید صورت می گیرد.شکل زیر نحوه انجام این فرایند را نشان می دهد:

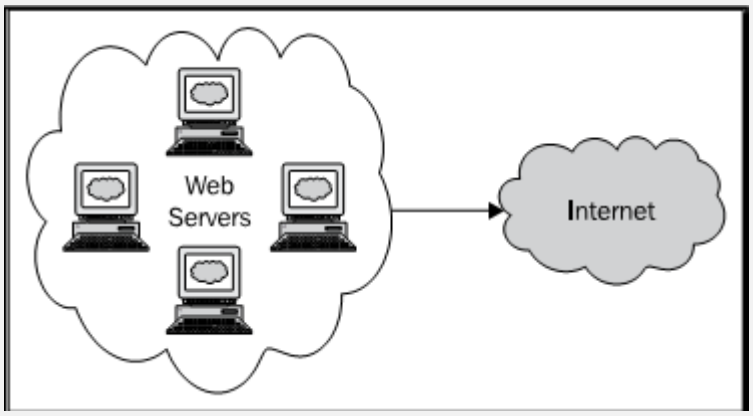


تصویر ۹-۱ موقعیت یک پراکسی سرور معمولی را در شبکه نشان می دهد.

همانطور که می بینید پراکسی سرور اسکوئید با ذخیره ی داده های موجود برروی وب سرورهای مختلف موجود در شبکه ی جهانی اینترنت و ارسال آن به کاربران در مواقع نیاز باعث افزایش کیفیت ارتباطات اینترنتی کاربران می شود.

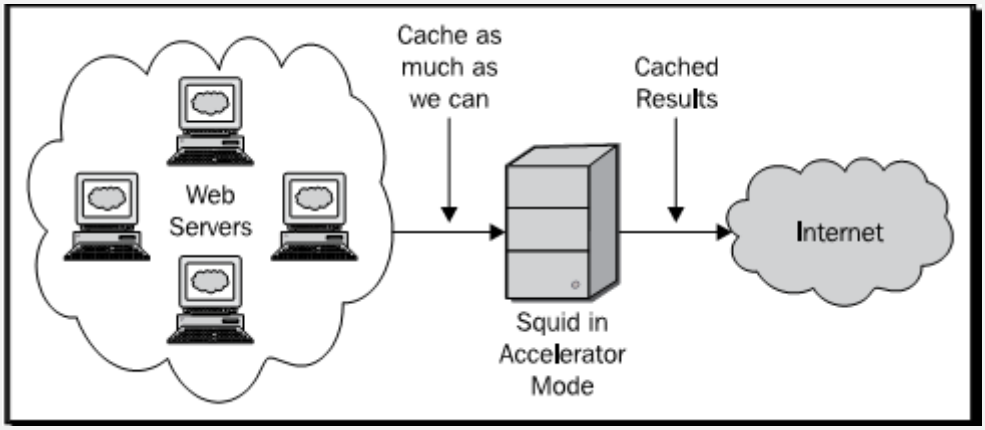
مروری بر پراکسی معکوس

برای درک بهتر این مطلب سناریویی را در نظر بگیرید که در آن وب سایت www.example.com برروی یک وب سرور قرار دارد و روزانه صدها و یا هزاران بازدید کننده محتوای موجود در آن را مشاهده می کنند در صورتکیه تعداد بازدید کنندگان آن از میزان ظرفیت و پهنای باند وب سروری که میزبانی وب سایت را برعهده دارد فراتر رود ،سرور با ترافیک و بار اضافی مواجه می شود و باعث توقف فعالیت و کاهش بازده آن یا به اصطلاح overload شدن آن می شود. برای جلوگیری از این مشکل با تقسیم بار ترافیکی میان چند وب سرور مجزا ،بار ترافیکی اعمال شده برروی هر یک ازآنها متعادل می شود شکل زیر این مطلب را به خوبی بیان می کند:



تصویر ۹-۲ چگونگی تقسیم بار میان وب سرورهای مختلف را به تصویر می کشد.

در این تصویر گروهی از وب سرورها میزبانی وب سایت example.com را برعهده دارند و درخواستهای بازدیدکنندگان در نقاط مختلف جهان را پاسخ می گویند.بسیاری از مطالبی که در یک وب سایت وجود دارد برا ی مدت زمان طولانی ای تغییر نمی کند مثلا قالب وب سایت،کدهای موجود در آن و سایر محتوا نظیر تصاویر ،آیکون ها و مقالات موجود در آن به ندرت دچار تغییر می شوند در این حالت استفاده از اسکوئید به صورت reverse proxy یا پراکسی معکوس باعث ذخیره سازی این محتوا و پاسخ گویی به طیف وسیعی از بازدیدکنندگان می شود همچنین اسکوئید با کاهش بار ترافیکی موجود به دلیل مراجعات زیاد به وب سرورها از overload شدن آنها جلوگیری می کند.تصویر زیر موقعیت اسکوئید را به همراه وب سرورها نشان می دهد:



تصویر ۹-۳ موقعیت نسبی اسکوئید به عنوان شتاب دهنده را نشان می دهد.

در اینجا اسکوئید دقیقا در مقابل وب سرورها قرار می گیرد و تمامی درخواستهایی که به سوی وب سرورها روانه می شوند

ابتدا از پراکسی سرور عبور کرده و در صورتکیه پاسخ درخواست موجود در مخزن پراکسی سرور وجود داشته باشد از همان جا پاسخ درخواست داده می شود و درخواست کاربران به سوی وب سرورها ارسال نمی شود و در صورتکیه پاسخ درخواستهای مورد نظر وجود نداشته باشد اسکوئید درخواست مورد نظر را به وب سرورهای اصلی میزبان وب سایت ارجاع می دهد به این شیوه به کارگیری اسکوئید به همراه وب سرورها ،حالت پراکسی معکوس ، شتاب دهنده و یا سرور جانشین نامیده می شود.حال در ادامه با نحوه پیکربندی اسکوئید جهت به کارگیری در این حالت آشنا می شویم.

پیکربندی اسکوئید در حالت سرور جانشین

به منظور به کارگیری اسکوئید در حالت surrogate یا سرور جانشین به همراه وب سرورها نیازمند استفاده از تنظیمات مناسب و متناسب با وضعیت موجود هستیم.می توان اسکوئید را به طور همزمان به عنوان سرور جانشین و ارجاع دهنده پیکربندی نمود که دراین حالت تنظیمات ACL ها باید با دقت فراوانی نوشته شوند.در صورتیکه وب سایت مورد نظر شما با ترافیک بسیار بالایی مواجه است توصیه می شود از دو سرور اسکوئید یکی به عنوان ارجاع دهنده و دیگر به عنوان جانشین استفاده کنید که در این حالت میزان کارایی افزایش می یابد.همچنین در صورتکیه اسکوئید به درگاه 80 جهت گوش کردن و دریافت درخواستها استفاده می کند وب سرور ما قادر به گوش کردن به همان آدرس ip اسکوئید نیست برای حل این مشکل سناریوهای زیر مطرح می شود:

- اسکوئید می تواند به پورت ۸۰ برروی یک IPعمومی گوش کند و وب سرور قادر به گوش کردن به پورت ۸۰ از طریق آدرس ۱۲۷.۰.۰.۱ یا همان loopback خواهد بود.
- وب سرور قادر به گوش کردن به پورت ۸۰ از طریق یک کارت شبکه ی مجازی (vlan) و از طریق آدرس های ip خصوصی خواهد .در صورتکیه وب سرور و پراکسی سرور اسکوئید هرکدام برروی سرورهای مجزایی قرار گیرند چنین مشکلاتی به طور کلی به وجود نخواهد آمد.

درگاه HTTP

پس از آشنایی با وضعیت عملکرد اسکوئید،مهمترین دستوری که می بایست پیکربندی شود http_port است که می بایست درگاه http به توسط تنظیمات مناسب و متناسب با شبکه پیکربندی شود.فرمت کلی http_port به منظور به کارگیری در حالت معکوس به صورت زیر است:

http_port 80 accel [options]

در اینجا باید آدرس درگاه مورد نظر مثلا ۸۰ مشخص شود و در کنار آن عبارت accel را قرار داده که این عبارت اسکوئید برروی درگاه ۸۰ در حالت معکوس قرار می گیرد هچنین در قسمت [options] سایر دستورهایی که به منظور پیکربندی

صحیح اسکوئید جهت ارتباط با وب سرورهای مورد نیاز است قرار می گیرد.

تنظیمات مربوط به HTTP در حالت پراکسی معکوس

در این قسمت به بررسی دستورهایی که همراه با http_port به کار می روند می پردازیم.

defaultsite

دستور (defaultsite=domain_name) defaultsite برای مشخص کردن نام دومین یا سایت و ساخت هدر میزبان به کار می رود نام دامنه مشخص شده در این دستور نام اصلی وب سایت است که کاربران از طریق مرورگر خود به آن دسترسی دارند.

Vhost

در صورت به کارگیری دستور vhost اسکوئید از دامنه هایی که از طریق میزبان های مجازی میزبانی می شوند پشتیبانی می کند.

Vport

برای فعال سازی میزبان مجازی بر پایه (IP-based virtual host support) ip باید از دستور vport استفاده کنیم.این دستور به دو طریق قابل استفاده است:

اگر دستور vport به کارگرفته شود اسکوئید از همان درگاه مربوط به هدر host استفاده می کند در صورتکیه در گاه مربوط به Host در دسترس نباشد،از درگاه http_port به منظور پشتیبانی میزبانی مجازی استفاده می کند.

در صورتیکه درگاه vport به صورت مجزا از طریق دستور vport=PORT_NUMBER مشخص شده باشد اسکوئید از شماره ی درگاه مشخص شده در قسمت PORT_NUMBER به جای http_port استفاده می کند در واقع در نوع دوم شماره ی درگاه اختصاصی برای این منظور درنظر گرفته می شود.

allow-direct

ارجاع مستقیم درخواستها در حالت پراکسی معکوس به صورت پیش فرض و به دلایل امنیتی غیر فعال است. در صورتکیه این امکان فعال باشد یک نفوذگر با استفاده از هدر های جعلی http و از طریق یک دامنه جعلی می تواند با ارسال درخواستهای متعدد به اسکوئید امکان به خطر انداختن شبکه را فراهم کند به دلیل حساسیت موجود برروی وب سرورها و محتوای آنها استفاده از دستورهایی که به نوعی ریسک حملات خرابکارانه را بالا می برد خودداری شود به هر حال با استفاده از دستور allow-direct می توان این امکان را فعال کرد.

Protocol

فرمان(protocol=STRING) protocol امکان بازتولید درخواستهای ارائه شده از سوی کاربران را فراهم می کند. مقدار پیش فرض این دستور HTTP است.

ignore-cc

در خواستهای HTTP موجود حامل هدر Cache-Control از طرف کاربران هستند که مشخص می کند یک درخواست مشخص دوباره بروزسانی و یا حذف شود.در صورت استفاده از دستور ignore-cc هدر های cache-control نادیده گرفته می شوند و اسکوئید پاسخ های کش شده را دوباره ذخیره می کند در صورتکیه هنوز زمان انقضای آنها باقی مانده باشد.

خطوط زیر نمونه ای از به کارگیری دستور http_port را نشان می دهند:

```
http_port 80 accel defaultsite=www.example.com
http_port 80 accel vhost
http_port 80 accel vport ignore-cc
```

درگاه HTTPS

فرض کنیم می خواهیم درخواستهایی را از طریق پروتکل HTTPS که به نوعی پروتکل HTTP با امنیت بیشتر محسوب می شود و درخواستهای کاربران را قبل از ارسال به صورت رمزنگاری شده در می آورد.برای اینکه اسکوئید بتواند این درخواستها را دریافت و به وب سرورها ارجاع دهد ابتدا آنها را از حالت رمزنگاری شده خارج کرده و سپس آنها را به سوی وب سرورهای مورد نظر ارسال می کند .

نکته: برای اینکه اسکوئید بتواند فرایند دریافت و ارجاع درخواستهای **HTTPS** *را انجام دهد باید ماژول* **ssl** *از طریق فرمان* **enable-ssl**—*در زمان کامپایل اسکوئید فعال شده باشد.*

تنظیمات HTTPS در حالت پراکسی معکوس

شکل کلی دستور https_port به صورت زیر است:

```
https_port [IP_ADDRESS:]port accel cert=certificate.pem [key=key.pem] [options]
```

در اینجا IP_ADDRESS به منظور تعیین آدرس ip اسکوئید به کار می رود این دستور اختیاری است و با مشخص نکردن آن نیز مشکلی پیش نمی آید.پارامتر port به منظور تعیین درگاهی که اسکوئید از طریق آن به درخواستهای HTTPS گوش می دهد به کار می رود.پارامتر certبرای تعیین مسیر کامل فایلهای ssl یا openssl و همچنین کلیدهای آن به کار می رود.پارامتر

key نیز یک عبارت اختیاری است و به منظور تعیین مسیر دقیق کلیدهای ssl به کار می رود و درصورتکیه این پارامتر تعیین نشود اسکوئید از همان مسیری گه توسط cert تعیین شده است استفاده می کند

نکته:برای اینکه بتوان از بسته های **openssl** *در اسکوئید استفاده کرد ابتدا باید از وجود و نصب کامل آن برروی سیستم عامل مورد نظر اطمینان حاصل نمائیم همچنین جهت دریافت و کسب اطلاعات بیشتر و دریافت مستندات کاملی در مورد سته های* **openssl** *به آدرس* **http://www.openssl.org** *مراجعه نمائید.*

defaultsite

دستور(defaultsite (defaultsite=domain_name برای مشخص کردن آدرس دامنه ا وب سایتی که قرار است اسکوئید درخواستهای HTTPS را به سوی وب سرورهای آن ارجاع دهد به کار می رود.

Vhost

مانند آنچه که در قسمت http_port بیان شد،دستور vhost به منظور پشتیبانی از میزبان های مجازی جهت استفاده از دامنه های مختلف به کار می رود. توجه داشته باشید در صورتکیه این ویژگی فعال باشد باید از پشتیبانی سیستم رمزنگاری و گواهی نامه های موجود از چند دامنه اطمینان حاصل نمائید.

Version

دستور (version=NUMBER) version به منظور تعیین نسخه ی SSL/TLS که از آن پشتیبانی می شود به کار می رود نسخه های پشتیبانی شده شامل موارد زیر است :

- Automatc detecton که نسخه ی مورد استفاده را به صورت خودکار شناسایی و تشخیص می دهد.
- SSLv2 only.
- SSLv3 only.
- TLSv1 only

Cipher

می توان فهرستی از رمزهای پشتیبانی شده به منظور استفاده در Openssl را از طریق دستور cipher (cipher=COLON_SEPARATED_LIST) اضافه نمود همچنین جهت کسب اطمینان از رمزهای مورد پشتیبانی در Openssl به آدرس http://www.openssl.org/docs/apps/ciphers.html مراجعه کنید و از رمزهایی که با نسخه ی جاری Opennssl مورد نظر شما سازگاری کامل دارند آشنا شوید.

Options

می توانیم گزینه های متوعی از تنظیمات مربوط به openssl را در فایل پیکربندی توسط دستور options (options=LIST) اضافه کنیم همچنین قبل از اعمال هر نوع تغییری از طازکاری این دستورات با کتابخانه ها و نسخه ی مورداستفاده ی open ssl اطمینان حاصل نمائید.با مراجعه به http://www.openssl.org/docs/ssl/SSL_CTX_set_options.html از engine-option هایی که توسط نسخه ی مورد استفاده ی شما پشتیبانی می شود آگاه شوید.

Clientca

دستور clientca (clientca=FILE) به منظور تعیین مسیر فهرستی از گواهی نامه های احراز هویت (Certfcate Authorites- CA) در مواردی که کاربری درخواست دریافت گواهی احراز هویت دارد را مشخص می کند.

Cafle

Cafile نیز به منظور اضافه کردن CA های جدید از طریق دستور (cafile=FILE) به کار می رود.

Capath

دستور capath (capath=DIRECTORY) برای اضافه کردن گواهی های احراز هویت CRL (Certfcate Revocaton List) به کار می رود که این فایل برخلاف موارد قبلی به منظور بررسی گواهی های امنیتی کاربران به کار می رود.

نکته:در صورتکیه از هیچ یک از گزینه های **clientca, cafile, or capath** *استفاده نشود ،اسکوئید به طور پیش فرض از کتابخانه های* **ssl** *موجود برروی سیستم عامل استفاده می کند.*

crlfle

crlfile (crlfile=FILE) برای تعیین مسیر دقیق تعدادی از CRL ها به کار می رود که به منظور بررسی گواهی های کاربران همانند مورد قبلی به کار می رود.

Sslfags

با استفاده از دستور sslflags (sslflags=LIST_OF_FLAGS)می توان یک یا چند flagکه برای مشخص کردن نوع ssl به کار رفته استفاده می شود به کار می رود.در ادامه گروهی از این flag ها را بررسی می کنیم.

NO_DEFAULT_CA

NO_DEFAULT_CA،CA های پیش فرضی که برپایه ی Openssl هستند را غیرفعال کرده و از آنها استفاده نمی کند.

NO_SESSION_REUSE

دستور NO_SESSION_REUSE ،تمامی اتصالات جدید ssl را به عنوان اتصالات مستقل فرض کرده و سیار ارتباطات ssl ی که قلا برقرار شده را دوباره به کار نمی گیرد.

VERIFY_CRL

در صورت به کارگیری دستور VERIFY_CRL ،فهرستی از CLR هایی که قبلا به وسیله دستورات crlfile و capath تعیین شده اند جهت بررسی گواهی های امنیتی کاربران استفاده می شود.

VERIFY_CRL_ALL

دستور VERIFY_CRL_ALL تمامی گواهی نامه های امنیتی کاربران که قبلا تعیین و ایجاد شده اند بررسی می شوند.

Sslcontext

sslcontext (sslcontext=ID) یک شماره ی ID برای شناسایی SSL به دستورات آن اضافه می کند و هرگاه پروتکل SSL برروی سرور مشغول فعالیت باشد از طریق این نام شناسایی عددی توسط سیستم عامل شناسایی شده و مانند سایر سرویس ها و برنامه های موجود در لینوکس و یونیکس با سیستم عامل ارتباط برقرار می کند.

Vport

Vport به منظور فعال سازی قابلیت پشتیبانی مجازی بر پایه ip برای شناسایی vport هنگام استفاده از دستور http_port به کار می رود .خطوط زیر نمونه ای تنظیمات پیکربندی مربوط به دستور https_port را نشان می دهد:

```
https_port 443 accel defaultsite=secure.example.com cert=/opt/squid/
etc/squid_combined.pem sslflags=NO_DEFAULT_CA
https_port 443 accel vhost cert=/opt/squid/etc/squid.pem key=/opt/
squid/etc/squid_key.pem
```

اضافه کردن وب سرورهای بخش مدیریت (Backend Webservers)

تاکنون روش های مختلف ارتباط اسکوئید با وب سرورها از طریق پروتکل های HTTP و HTTPS را آموختیم زمانی که اسکوئید یک درخواست از نوع HTTP یا HTTPS را دریافت می کند در صورتکیه نتواند با استفاده از داده های موجود در مخزن خود پاسخ درخواست را پیدا کند درخواست مورد نظر را ابتدا به وب سرور حاوی پاسخ این درخواست ارجاع داده و سپس آن را برای کاربر درخواست کننده ارسال می کند.استفاده از چند وب سرور با پهانی باند کافی و اضافه کردن آن در فایل

پیکربندی باعث سرعت بخشیدن به این فرایند و در نتیجه رضایت بازدید کنندگان از وب سایت خواهد شد.با استفاده از دستور cache_peer می توان یک یا چند وب سرور را به تنظیمات پیکربندی اسکوئید اضافه نمود.

تنظیمات کش سرورهای همکار در حالت پراکسی معکوس

گزینه هایی که برای استفاده از دستور cache_peer در حالت پراکسی معکوس ایجاد شده اند شامل موارد زیر است.

Originserver

در صورت به کارگیری دستور originserver همراه با cache_peer ،اسکوئید با وب سرور تعریف شده در آن مانند وب سرور اصلی رفتار می کند.

Forcedomain

دستور forcedomain (forcedomain=domain_name) برای تعیین نام میزبانی که اسکوئید همواره از آن به عنوان هدر میزبان به همراه نام دامنه استفاده می کند این دستور به منظور حل مشکل وب سرورهایی که برروی چند دامنه با یکدیگر ارتباط دارند به کار می رود.در صورتکیه وب سرورهای موجود از قابلیت چند دامنه ای پشتیبانی می کند بایداز به کارگیر ی این دستور خودداری شود.مثال زیر چگونگی به کارگیری این دستورات را به همراه cache_peer به خوبی نشان می دهد:

```
cache_peer 127.0.0.1 parent 80 0 no-query no-digest originserver
cache_peer local_ip_of_web_server parent 80 no-query originserver
forcedomain=www.example.com
```

آشنایی با پروتکل surrogate

در ادامه با چگونگی کارکرد پروتکل **surrogate** و ارتباط آن با هدر های HTTP آشنا می شویم.

- هنگامی که یک وب سرور از نوع surrogate درخواستی را دریافت می کند هدر درخواست مورد نظر را به صورت زیر در آورد:

```
GET / HTTP/1.1
...
Surrogate-Capability: mirror.example.com="Surrogate/1.0"
```

...

- زمانی که درخواستی از یک سرور surrogate دریافت می شود،وب سرور پاسخ مناسب درخواست را به همراه هدر http متناسب با آن را به سوی اسکوئید ارسال می کندوخطوط زیر نمونه ای از هدر فایلهای surrogate را نشان می دهد:

HTTP/1.1 200OK

...

Cache-Control: no-cache, max-age=1800, s-max-age=3600

Surrogate-Control: max-age=43200;mirror.example.com

...

کاربران نهایی می تواند پاسخ هایی درفات شده را به مدت نیم ساعت نگهداری کنند که این مقدار توسط دستور Cache-Control:max-age=1800 تعیین می شود.پراکسی سرورهای موجود در شبکه نیز می توانند درخواست موجود به مدت یک ساعت نگهداری کنند این مقدار نیز توسط دستور Cache-Control:s-max-age=3600 تعیین شده است وب سرورهای جانشین (surrogate) که توسط آدرس mirror.example.com شناسایی می شوند قادر به نگهداری درخواست مذکور به مدت یک نیم روز خواهند بود کهاین مقیزان نیز توسط دستور Surrogate-Control: max-age=43200 تعیین می شود.همانطور که می بینیم در داخل هدر های HTTP میزان زمان نگهداری درخواستها و همچنین اعمال کنترلی لازم برای نگهداری پاسخ ها برای یک مدت زمان مشخص برای کاربران و سایر سرورهای پراکسی تعریف شده است .اطلاعات بیشتر در باره ی پروتکل surrotage در http://www.w3.org/TR/edge-arch موجود است.

تنظیمات پیکربندی مورد نیاز برای پشتیبانی از پروتکل surrotage

در فایل تنظیمات پیکربندی اسکوئید دو دستور در ارتباط با پروتکل surrotage ایجاد شده است که در ادامه به هریک آشنا می شویم:

httpd_accel_surrogate_id

تمامی سرورهایی که از پروتکل suttotage استفاده می کنند به یک نام شناسایی جهت ارتباط با سرورهای اصلی نیاز دارند این نام های شناسایی می تواند منحصر به فرد باشدو. با درمیان سایر رورهای پراکسی که به صورت گروهی فعالیت می کنند به اشتراک گذاشته شود که این موضوع به نوع تنظیمات اعمالی برروی gateway بستگی

دارد.مقدار پیش گزیده برای این نام شناسایی استفاده از مقداری است که در دستور visible_hotname به کار رفته است که برای تغییر این مقدار می توان از دستور httpd_accel_surrogate_id مانند مثال زیر استفاده کرد:

```
httpd_accel_surrogate_id mirror1.example.com
```

این خط از تنظیمات پیکربندی ،نام شناسایی را به mirror1.example.com تغییر می دهد.

```
httpd_accel_surrogate_remote
```

سرورهای راه دور از نوع surrogate مانند Content Delivery Network or CDN ،در هدر درخواستهای HTTP ای که از طرف آنها ارسال می شود شامل هدر ی به نام Surrogate-Control: no-store-remote است به این معنی که پاسخهای حامل این کد نباید در کش ذخیره شوند و مستقیما باید به منشا درخواست که معمولا برروی وب سرورهای اصلی قرار دارند ارجاع داده شوند .با استفاده از دستور http_accel_surrogate_remote می توان پراکسی سرور خود را به عنوان یک سرور نوع surrogate آگاه کرد که مقدار این دستور باید مانند خط زیر باشد:

```
http_accel_surrogate_remote on
```

استفاده از این دستور زمانی توصیه می شود که پراکسی سرور موردنظر دو یا بیش از دو گره از سرور اصلی فاصله داشته باشد.

ثبت گزارشات رخداد ها به روش گزارش گیری وب سرورها

هنگام استفاده از اسکوئید در حالت پراکسی معکوس ، بسیاری از اطلاعات پیام های ثبت رخداد وب سرورها دراثر عدم همخوانی با فرمت موجود در سیستم گزارش گیری اسکوئید از بین می روند .دلیل این مشکل متفاوت بودن چگونگی ثبت پیام های ثبت رخداد در فایل access.log اسکوئید و فایل ثبت پیام های رخداد وب سرورها است برای حل این مشکل باید نحوه ثبت پیام های رخداد در فایل access.log اسکوئید نیز مشابه فایل ثبت رویداد در وب سرورها باشد.با استفاده ازدستور common همراه با access_log بسیاری از جزئیات موجود در access.log حذف شده و نحوه ثبت رویدادهای اسکوئید نیز مانند فایلهای ثبت رویداد وب سروها ثبت و ذخیره می شود.

نادیده گرفتن بروزرسانی صفحات در مرورگرها

بسیاری از مرورگرهای موجود دارای یک دکمه به نام reload هستند که در صورت استفاده از آن هدر cache-control را به no-cache تغییر می دهد که این موضوع سبب می شود اسکوئید داده مورد نظر را از مخزن کش حذف کرده و درخواست مورد نظر را مجددا ازسرور اصلی آن دریافت کند درحالیکه داده کش شده ی قبلی نیز هنوز دارای اعتبار بوده است این موضوع در دراز مدت سبب از بین رفتن تدریجی منابع موجود در مخزن کش سرور و استفاده ی بیش از اندازه از پهنای باند

خواهد شد.برای حل این مشکل ۳ راه حل وجود دارد و آن استفده از دستورات خاصی همراه با http_port و refresh_pattern ها می باشد البته توجه داشته باشید دستور refresh_pattern تنظیمات خود را برروی سرور و کاربر به طور همزمان اعمال می کند که در صورت استفاده ی نابجا باعث دریافت اطلاعات قدیمی و نادرست توسط کاربر شود.

ignore-cc

با دستور ignore-cc در بخش تنظیمات http_port آشنا شدیم در صورت استفاده از این دستور ،اسکوئید هدر cache-control را از جانب کاربران به طو رکلی نادیده می گیرد .مثال زیر چگونگی به کارگیری آن را بیان می کند:

```
http_port 80 accel defaultsite=example.com vhost ignore-cc
```

ignore-reload

به کارگیری ignore-reload همراه با دستور refresh_psttern باعث نادیده گرفتن بروزرسانی های مرورگرها و استفاده از اطلاعات موجود در مخزن کش به منظور پاسخگویی به درخواستها می شود به هر حال این دستور از طرفی موجب کاهش مراجعه با سرورها خارجی و از طرفی ممکن است موجب دریافت اطلاعات قدیمی و خارج از منبع توسط کاربر شود.مثال زیر چگونگی استفاده از این دستور را نشان می دهد:

```
refresh_pattern .   0 20% 4320 ignore-reload
```

reload-into-ims

دستور ignore-reload در مواردی باعث ایجاد مشکلاتی از جمله مشاهده محتوای منقضی شده می شود که این موضوع به خصوص در مورد سایتهای خبری بیشتر مشهود می شود و بعضا سبب ایجاد نارضایتی نیز می شود.دستور reload-into-ims درواقع به نوعی جدیدترین نسخه ی داده کش شده را در اختیار کاربر قرار می دهد و موجب کاهش اتلاف پهنای باند هنگام یکسان سازی و بررسی اعتبار داده ها می شود .برای مثال:

```
refresh_pattern .   0 20% 4320 reload-into-ims
```

به کارگیری Access Control در حالت پراکسی معکوس

تاکنون در مورد ارتباطات اسکوئید با سایر وب سرورها و برقراری ارتباط در حالت Reverse Proxy مطالب مختلفی آموختیم اما تا این لحظه کلیه ی ارتباطات ما با وب سرورها کاملا آزاد و بدون هیچ گونه محدودیتی بوده است در واقع ابزار لازم برای کنترل IP های ورودی و خروجی،کاربران بازدید کننده و سرورها را اختیار نداشته ایم .Access Control ها که در فصلهای قبل با آن آشنا شدیم در مورد این ارتباطات نیز کارساز هستند.در ادامه با این دستورها بیشتر آشنا می شویم.

به کارگیری اسکوئید فقط درحالت پراکسی معکوس

هنگامی که از اسکوئید فقط به عنوان پراکسی معکوس استفاده می کنیم باید از دسترسی کاربران به طور مستقیم . بدون واسطه با سرورهای اصلی ای که اسکوئید با آنها در ارتباط است جلوگیری کنیم در مثال زیر سرورهای www.example.com و www.example.net با پراکسی سرور اسکوئید با توجه به تنظیمات زیر در ارتباط هستند:

```
acl orign_servers dstdomain www.example.com www.example.net
http_access allow origin_servers
http_access deny all
```

با استفاده از این تنظیمات کلیه ی درخواستهایی که منبع آنها www.example.com و www.example.net است توسط اسکوئید پذیرفته می شوند.

اسکوئید در حالت پراکسی معکوس و ارجاع دهنده

زمانی که از اسکوئید به طور همزمان به صورت ارجاع دهنده و پراکسی معکوس استفاده می کنیم ،هنگام طراحی سطح دسترسی ها (Access Controls)باید موراد زیر را به خاطر داشته باشیم:

- کاربرانی که از پراکسی سرور به عنوان ارجاع دهنده (forwarder) استفاده می کنند باید به تمامی وب سایتها به جز آنهایی که توسط خود ما محدودشده اند دسترسی کامل داشته باشند.
- اسکوئید باید تمامی درخواستهای کاربرانی که مقصد آنها وب سرورهایی که اسکوئید با آن ها در ارتباط است را بپذیرد به جز کاربرانی که توسط خود ما در لیست سیاه قرار داده شده اند.

فرض کنیم کاربران موجود در شبکه که از پراکسی سرور ما استفاده می کنند در محدوده ی آدرس 192.0.2.0/24 قرار دارند و به تمامی وب سایتها به جزء www.example.net دسترسی دارند تنظیمات دسترسی مطابق زیر نوشته می شوند:

```
acl our_clients src 192.0.2.0/24
acl blacklisted_websites dstdomain www.example.net
http_access allow our_clients !blacklisted_websites
```

```
http_access deny all
```

حالا فرض کنیم در شبکه ای که از پراکسی سروراسکوئید در آن به منظور افزایش سرعت دسترسی به وب سایت www.example.com به کار می رود گروهی از بازدیدکنندگانی که در محدوده ی آدرس ۲۰۳.۰.۱۱۳.۰/۲۴ قرار دارند فعالیت های مشکوک به خرابکاری و نفوذ از خود نشان می دهند به همین دلیل مایل به دسترسی این گروه از کاربران به وب سایت مورد نظر نیستیم برای جلوگیری ازدسترسی آنها تنظیمات دسترسی به صورت زیر نوشته می شوند:

```
acl origin_servers dstdomain www.example.com
acl bad_visitors src 203.0.113.0/24
http_access allow origin_servers !bad_visitors
http_access deny all
```

می توان تنظیمات پیکربندی مطرح شده در این دو بخش را به صورت زیر با هم ترکیب کرد:

```
# ACLs for Forward Proxy
acl our_clients src 192.0.2.0/24
acl blacklisted_websites dstdomain www.example.net
# ACLs for Reverse Proxy
acl origin_servers dstdomain www.example.com
acl bad_visitors src 203.0.113.0/24
# Allow local clients to access allowed websites
http_access allow our_clients !blacklisted_websites
# Allow visitors to access origin servers
http_access allow origin_servers !bad_visitors
# Deny access to everyone else
http_access deny all
```

تنظیمات مطرح شده دسترسی به تمامی محتوای موجود در وب به جز www.example.net را فراهم می کند همچنین تمامی بازدیدکنندگان جز کاربرانی که در محدوده ی آدرس ۲۰۳.۰.۱۱۳.۰/۲۴ قرار دارند اجازه بازدید از وب سایت www.example.com را خواهند داشت.

ارائه ی چند مثال از تنظیمات پیکربندی

در اینجا چند مثال از تنظیمات مربوط به اسکوئیددر حالت پراکسی معکوس ارائه می شود.

قرارگیری وب سرور و اسکوئید برروی یک سرور

در این مثال پراکسی سرور اسکوئید به منظور افزایش سرعت دستری به وب سایت مورد نظر به همراه وب سرور برروی یک سرور راه اندازی شده اند بدین منظور باید از دسترسب وب سرور به آدرس ۱۲۷.۰.۰.۱ و درگاه ۸۰ اطمینان حاصل نمائیم در این صورت تنظیمات پیکربندی به صورت زیر خواهد بود:

```
http_port 192.0.2.25:80 accel defaultsite=www.example.com
cache_peer 127.0.0.1 parent 80 0 no-query originserver name=example
cache_peer_domain example .exmaple.com
```

در اولین خط از تنظیمات پیکربندی،اسکوئید از طریق آدرس ۱۹۲.۰.۲.۲۵ برروی درگاه ۸۰ به درخواستهای کاربران گوش می کند و به منظور افزایش سرعت وب سایت مورد نظر درخواستهای کاربران ی که از آدرس example.com مراجعه کنند پاسخ می دهد در خط دوم وب سرور موردنظر از طریق آدرس ۱۲۷.۰.۰.۱ برروی درگاه ۸۰ به درخواستهای موجود گوش می کند در آخزین خط نیز cache peer تعریف شده فقط درخواستهایی که مربوط به آدرس example.com و زیردامنه های آن است پاسخ می دهد.

توزیع بار برروی چند سرور

دراین مثال سه وب سرور با آدرس های ۱۹۲.۰.۲.۲۵، ۱۹۲.۰.۲.۲۶، ۱۹۲.۰.۲.۲۷وجود دارند که هر سه به منظور میزبانی وب سایت www.example.com به کار می روند تمام وب سرورها به پورت ۸۰ گوش می دهند و اسکوئید برروی یک سرور مجاز واز طریق ip عمومی با این سرورها در ارتباط است بدین ترتیب تنظیمات پیکربندی برای این مثال به صورت زیر است:

```
http_port 80 accel defaultsite=www.example.com
```

```
cache_peer 192.0.2.25 parent 80 0 no-query originserver round-robin
name=server1
cache_peer 192.0.2.26 parent 80 0 no-query originserver round-robin
name=server2
cache_peer 192.0.2.27 parent 80 0 no-query originserver round-robin
name=server3
cache_peer_domain server1 .example.com
cache_peer_domain server2 .example.com
cache_peer_domain server3 .example.com
```

به کرگیری دستور round-robin در این تنظیمات موجب توزیع بار(load balance)برروی وب سرورها خواهد شد.

ایجاد شتاب در پاسخگویی وب سرورهایی که میزبانی چند وب سایت را برعهده دارند

در این مثال وب سایت example.com و زیر دامنه های آن برروی یک سرور به آدرس ۱۹۲.۰.۲.۲۵، example.net و زیر دامنه های آن برروی سرور دیگری به آدرس ۱۹۲.۰.۲.۲۶ و همچنین example.org برروی یک سرور مجزای دیگر به آدرس ۱۹۲.۰.۲.۲۷میزبانی می شوند و در آخر نیز پراکسی سرور اسکوئید برروی یک سرور مجزای دیگر در نقطه ی میانی این ارتباطات قرار دارد و با این سرورها در ارتباط است مانند موارد قبل اسکوئید در اینجا نیز نقش کش کننده ی اطلاعات و به مظنور افزایش سرعت دسترسی به این وب سایتها به کار می رود،تنظیمات پیکربندی مورد نیاز برای این سناریو به صورت زیر است:

```
http_port 80 accel vhost defaultsite=www.example.com ignore-cc
cache_peer 192.0.2.25 parent 80 0 no-query originserver name=server1
cache_peer 192.0.2.26 parent 80 0 no-query originserver name=server2
cache_peer 192.0.2.27 parent 80 0 no-query originserver name=server3
cache_peer_domain server1 .example.com
cache_peer_domain server2 .example.net
cache_peer_domain server3 .example.org
```

به دلیل وجود آدرس های گوناگون و در نتیجه درخواستهای متفاوتی که به سوی هر وب سرور ارسال می شود در اینجا مجاز به استفاده از دستور round-robin نیستیم همچنین دستور cache_peer_domain نیز به منظور افزایش امنیت به کاررفته است.

خلاصه فصل

دراین فصل مطالب بسیار متنوعی درمورد پیکربندی اسکوئید در حالت پراکسی معکوس مطرح شد این روش که هدف اصلی آن افزایش سرعت پاسخگویی وب سایت ها به بازدیدکنندگان آن است با استفاده از فرایند کش شدن اطلاعات و همچنین توزیع بار میان وب سرورهای گوناگون از مشکلاتی چون overload شدن وب سرورها و درنتیجه از دسترسی خارج شدن وب سایت جلوگیری می کند به طور کلی مطالب ارائه شده در این فصل را می توان در موارد زیر خلاصه کرد:

- چگونگی افزایش سرعت دسترسی به محتوای وب سایت از طریق به کارگیری اسکوئید
- پیکربندی و ارتباط اسکوئید با وب سرورها به منظور کش کردن اطلاعت و کاهش ترافیک وب سرورها
- استفاده از سرورهای جانشین به اسکوئید منظور ارجاع صحیح درخواستها به وب سرورها
- ارائه مثال های متنوع در مورد چگونگی ارتباط اسکوئید و وب سرور ها در سناریو های مختلف

در فصل آتی درمورد پیکربندی اسکوئید در حالت Intercept یا شفاف مطالبی عنوان خواهد شد.

فصل دهم

پیکربندی اسکوئید در حالت شفاف

در این فصل در مورد چگونگی راه اندازی و پیکربندی اسکوئید در حالت شفاف یا transparent و تنظیمات مربوط به آن مطالبی عنوان خواهد شد.

مطالب عنوان شده در این فصل شامل موارد زیر است:

- آشنایی با کش سرور در حالت شفاف
- مزایای استفاده از اسکوئید در حالت شفاف
- مشکلات کش سرور در حالت شفاف
- انتقال ترافیک پروتکل HTTPبه سوی اسکوئید
- راه اندازی کش سرور در حالت شفاف

آشنایی با کش سرور در حالت شفاف

در شبکه های بزرگ که صدها و یا هزاران کاربر ثابت و متغیر که از سرویس های مختلف اینترنت استفاده می کنند اعمال تنظیمات پراکسی برروی مرورگرهای تمامی کاربران بسیار مشکل و زمان بر است.Intercept Cacing و یا همان کش سرور در حالت شفاف کلید حل این مشکل است ،این روش که به نامهایی همچون Transparent Caching و یا Cache Redirection نیز معروف است ترافیک موجود در شبکه برروی کش سرور مورد نظر هدایت شده و تمامی کاربران بدون نیاز به هر گونه اعمال تنظیمات پراکسی به کش سرور دسترسی خواهند داشت .در واقع این روش همان گونه که از نامش پیداست درخواستهای کاربران بدون آنکه خود متوجه باشند از کش سرور عبورمی کند.برای استفاده از کش سرور در حالت شفاف معمولا به یک مسیریاب یا سوییچ در شبکه جهت هدایت ترافیک به سوی کش سرور اسکوئید مورد نیاز است در واقع مسیریاب یا سوییچ موجود در شبکه ترافیک کاربران شبکه را جهت عبور از کش به سوی کش سرور هدایت می کند علاوه بر این به یک ابزار Packet Filtering مانند Netfilter و iptable در سیستم عامل جهت کنترل و هدایت ترافیک به کاربران مورد نیاز خواهد بود.مراحل زیر به ترتیب چگونگی عملکرد کش سرور در حالت شفاف رابیان می کنند:

۱. یک کاربر آدرس http://www.example.com/index.html را درخواست می کند.

۲. اولین کار تبدیل نام دامنه مودر نظر به آدرس ip آن است که این مرحله ترجمه آدرس دامنه به IP از طریق یک DNS سرور صورت می گیرد که در اینجا این آدرس ۱۹۲.۰.۲.۱۰می باشد.

۳. پس از ترجمه نام دامنه ، کاربر از طریق درگاه 80 و پروتکل TCP/IP با آدرس ۱۹۲.۰.۲.۱۰ارتباط برقرار می کند.

۴. ارتباط برقرار شده در مرحله ۳ از طریق یک مسیریاب یا سوییچ به سوی پراکسی سرور اسکوئید هدایت می شود در حالت معمول پس از برقراری ارتباط کاربر با سرورهای اصلی و خارجی به تبادل داده می پردازند.

۵. برروی پراکسی سرور اسکوئید،Packet های ارسالی از طریق کاربران توسط ابزارهای Packet Filtering از طریق درگاه ۸۰ دریافت و ازطریق درگاهی دیگر مثلا ۳۱۲۸ به سوی کاربران ارسال می شود.

۶. سرانجام اسکوئید داده های دریافت شده را مانند یک وب سرور به کاربران از طریق برقراری ارتباط TCP با کاربر مورد نظر ارسال می کند.

۷. در مرحله ی قبل کاربری که به تبادل اطلاعات با اسکوئید می پردازید احساس می کند که باید یک سرور خارجی در ارتباط است در حالی که با پراکسی سرور اسکوئید ارتباط برقرار کرده است بنابراین یک کاربر معمولی هیچ تفاوتی میان سرور

اسکوئید و یک سرور خارج از شبکه احساس نمی کند.

۸ پس از برقراری ارتباط میان کاربر و پراکسی سرور،درخواست های HTTP مورد نظر به سوی پراکسی سرور فرستاده می شود.

۹. پس از دریافت درخواست کاربر اسکوئید به بررسی درخواست مورد نظر می پردازد و درصورتکیه پاسخ درخواست در کش سرور موجود باشد اسکوئید از طریق آن پاسخ درخواست را ارسال می کند و در غیر اینصورت درخواست را به سوی سرور اصلی آن جهت پاسخگویی مستقیم ارسال و پس از دریافت پاسخ به کاربر مورد نظر تحویل می دهد.توجه داشه باشید کلیه ی اینمراحل در مدت زمان بسیاری کوتاهی صورت می پذیرد و درواقع هرچه مدت زمان انجام این مراحل کوتاه تر باشد سرعت دریافت و ارسال داده میان کاربران و اسکوئید سریعتر و در نتیجه بازده شبکه بالاتر می رود.

مزایای استفاده از کش سرور در حالت شفاف

مواردی که در ادامه مطرح خواهند شد تعدادی از مزایای عمده اسکوئید در حالت شفاف را بیان می کند.

عدم نیاز به انجام تنظیمات برروی کاربران

درواقع شاید مهمترین دلیلی که بتوان برای استفاده از حالت شفاف بتوان بیان کرد ،عدم نیاز به هیچ گونه تنظیمات برروی کاربران است در این روش تمامی درخواستهای کاربران از طریق سوییچ و یا مسیریاب موجود در شبکه به سوی کش سرور هدایت می شود همانظور که در ابتدا هم اشاره شد در یک سازمان و یا شرکت که باهزاران کاربر ، انجام تنظیمات برروی تمامی یکایک آنها امری غیر ممکن و یا بسیار دشوار است در حالیکه در حالت شفاف تمامی کاربران شبکه بدون آنکه خودمتوجه باشند از کش سرور استفاده می کنند.

افزایش قابلیت اطمینان و امنیت

یک مسیریاب و یا سوییچ بدون نیاز به پراکسی سرور نیز می تواند درخواستهای کاربران را به سوی مقصد بدون هیچ گونه مشکلی هدایت نماید.اما با استفاده از اسکوئید در حالت شفاف می توان تمامی ترافیک شبکه را ازطریق ابزارهایی چون دیوار آتش و redirector ها کنترل و مدیریت کرد این درحالی است انجام بسیاری از این اعمال از طریق سایر ابزارها مانند مسیریابها و سوییچ ها بسیار مشکل و یا غیرممکن است همچنین قابلیت کش کردن اطلاعات به نوبه خود موجب افزایش کیفیت و راندمان شبکه خواهد شد.دلیل دیگری که برای استفاده از سرورپراکسی می توان بیان کرد،مخفی نگه داشتن اطلاعات کلیدی کاربران مانند IP حقیقی آنها و سایر مشخصات و درنتیجه کاهش خطرات مربوط به امنیت کاربران عنوان نمود.

معایب کش سرور در حالت شفاف

هر چند استفاده از کش سرور در حالت شفاف دارای مزایای بسیاری است اما بالطبع معایبی را نیز به همراه دارد از جمله نقض شدن بعضی از استاندارد های موجود در TCP/IP اشاره کرد موارد زیر چند نمونه از مهمترین معایب کش سرور در حالت شفاف را بیان می کند:

نقض استاندارد های TCP/IP

وظیفه ی اصلی یک مسیریاب یا سوئیچ در شبکه ،هدایت درخواستها و داده های کاربران به سوی سرور های مقصد است.طبیعی است انتقال این درخواستها به سوی سرور دیگری از جمله یک پراکسی سرور موجب نقض استاندارد های کلی پروتکل TCP/IP می شود از طرف دیگر پراکسی سرورها درخواستها و داده هایی را دریافت می کنند که مقصد اصلی آنها یک پراکسی سرور نیست که این خود باز هم نقض آشکار استاندارد های موجود در شبکه به شمار می رود از طرف دیگر معمولا نوع سیستم عاملی که کاربران و پراکسی سرور از آن استفاده می کند متفاوت است و این موضوع از یک ارتباط منسجم و دو طرفه میان سیستم عاملها که به طور حتم از مکانیسم های خاص خود در انتقال و دریافت اطلاعات استفاده می کنند جلوگیری می کند البته این بدان بعنی نیست که ارتباطات دچار مشکل می شود اما قطعا ناراسایی هایی در این ارتباطات به چشم می خورد که گرچه از دید یک کاربر عادی کاملا پنهان است اما یک متخصص شبکه این موضوع را به خوبی احساس می کند.

مشکلات مسیریابی

به دلیل خروج داده های موجود در شبکه از مسیر اصلی و انتقال آنها به پراکسی سرور مشکلاتی از قبیل اختلال در مشاهده وب سایتها محتمل به نظر می رسد البته باید توجه داشت بسیاری از این مشکلات از وجود نقص در پیکربندی اسکوئید ناشی می شود اما به هر حال بخشی از این مشکلات نیز ناشی از انحراف جریان اطلاعات از مسیر اصلی آن است.مسیریابهای شبکه ترافیک را از طریق مسیری که برای آنها تعریف شده به سوی پراکسی سرور ارسال می کنند حال اگر برخی از درخواستها از مسیری غیر از آنچه تعریف شده است عبور کنند درخواست مورد نظر از پراکسی سرور عبور نمی کند و مشکلاتی از قبیل قطع ارتباط و اشکال در نمایش وب سایتها از جمله ی این مشکلات است البته باز هم تاکید می شود بسیاری از این مشکلات ناشی از ضعف پیکربندی موجود در اسکوئید و یا مسیریابهای شبکه می باشد که قبل از هر چیز باید از وجود یک پیکربندی مناسب برروی دستگاههای شبکه اطمینان حاصل کرد.

عدم وجود سیستم احراز هویت

ابزارهای احراز هویت که تعدادی از آنها در این کتاب معرفی شدند برای اینکه بتوانند کاربران را از طریق نام کاربر ی و یا

رمز عبور و یا سایر روش ها احراز هویت کنند ، باید برروی مرورگر کاربران تنظیمات پراکسی سرور انجام گیرد در غیر اینصورت وب سرورها قادر به تشخیص کاربران مختلف نیستند دلیل این کار آن است هنگامی که کش سرور درحالت شفاف پیکربندی شده باشد ،تمامی درخواستهای کاربران از طریق یک آدرس ip و در نتیجه مانند یک کاربر معمولی به سوی وب سرورها ارسال می شود که البته این موضوع خود موجب افزایش امنیت کاربران می شود اما از طرفی باعث هدم کارکرد ابزارهای تشخیص هویتی که عملکرد آنها براساس آدرسهای مختلف ip است می شود.

عدم پشتیبانی از سایر پروتکلهای ارتباطی

یک پراکسی سرور در حالت شفاف فقط قادر به تشخیص درخواستهایی است که برپایه پروتکل HTTP و از شناسایی هدر های HTTP ارسال می شود می باشد به عبارت دیگر سایر پروتکل های موجود مانند FTP قادر به استفاده از حالت کش سرور در حالت شفاف نیستند البته این بدان معنا نیست که برقراری ارتباط از طریق آنها ممکن نیست بلکه درخواستهای و داده های مورد نظر از کش سرور عبور نمی کنند.

آشکار شدن اطلاعات کاربران

در بخش مزایای کش سرور در حالت شفاف یکی از مزایایی که برای این حالت بیان شد ،پنهان سازی اطلاعات کاربران در مقابل خرابکارن اینترنتی بود اما همانطور که گفته شد اسکوسید فقط قادر به دریافت اطلاعات از طریق پروتکل HTTP در حالت شفاف است و سایر پروتکل های ضرور ی ارتباطی مانند FTP،DNS و نظیر آنها قادر به عبور از کش سرور نیستند و کاربران هنگام استفاده از این پروتکل ها به طور کاملا آشکار و با مشخصات حقیقی خود در معرض دید قرار می گیرند که این موضوع در برخی از شبکه های حساس چندان مطلوب به نظر نمی رسد.

عدم امکان اعمال محدودیت بر آدرس های IP

به دلیل نوع عملکرد کش سرور در حالت شفاف،امکان اعمال محدودیت بر آدرس های ip در بسیاری از موارد وجود ندارد و باید محدودیت های مورد نظر از طریق سایر دستگاههای شبکه مانند سوئیچ ها و مسیریاب ها صورت بگیرد.البته این امر به میزان بسیار زیادی به نوع طراحی و شرایط شبکه نیز بستگی دارد مثلا در صورتکیه کاربران از آدرسهای معتبر اینترنتی و یا آدرسهای مشخص برای هر کاربر استفاده کنند امکان اعمال محدودیت از طریق ACL ها وجود دارد.

پشتیبانی از پروتکل ها

یکی از مسائلی که همواره در این زمینه مطرح است،بحث عدم پشتیبانی نسخه های قدیمی اسکوئید از پروتکلهای جدید است همچنین عملکرد اسکوئید به طور کامل به دریافت و شناسایی هدر های HTTP بستگی د ارد و درصورتکیه یک هدر جدید که برای اسکوئید قابل فهم نیست دریافت شود در این صورت اسکوئید هیچ گونه راهکاری برای پاسخگویی و

رفتار با آن را ندارد البته این موضوع ممکن است در سایر نرم افزارها نیز وجودداشته باشد و یک مشکل محسوب نشود و طبیعی است که ابزارها و پروتکلهای نوظهور در نرم افزارهای قدیمی مورد پشتیانی قرار نگیرند اما به هر حال این موضوعی است که همواره باید مورد توجه قرار گیرد و تنها راه حل ان نیز بروزرسانی مداوم اسکوئید در مواردی که تغییراتی در پروتکلهای موردپشتیبانی اسکوئید به وجود می آید.

مشکلات امنیتی

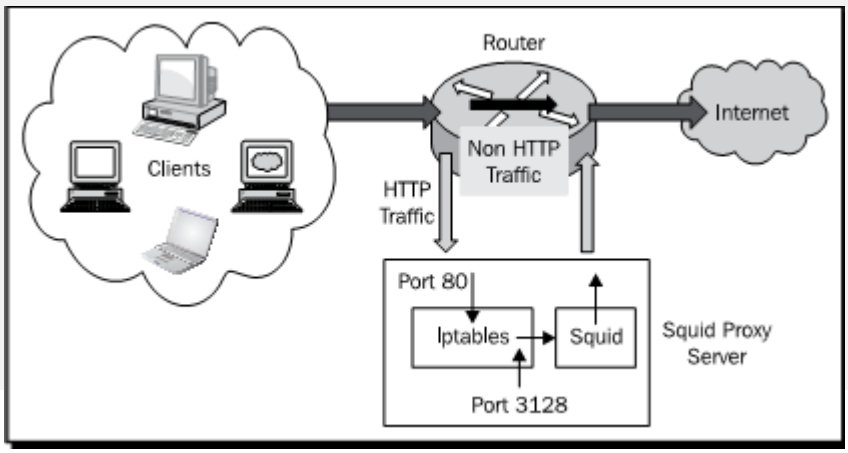
همانطور که گفته شد برای اینکه اسکوئید بتواند درخواستهای مختلف را شناسایی و تشخیص دهد ،از هدر های HTTP استفاده می کند و این درحالی است که یک نفوذگر به راحتی قادر به شبیه سازی این هدر ها و ارسال درخواستهای آلوده به انواع بدافزارها و کدهای مخرب به سوی اسکوئید است که ممکن است باعث ایجاد آلودگی در سطح کاربران شبکه شود.تا اینجا انواع مزایا و معایب استفاده از حالت شفاف در کش سرورها را بررسی کردیم حال با توجه به میزان مزایا و معایبی که این روش در اختیار شما می گذارد و همچنین سطح انتظارات شما تصمیم گیری در مورد استفاده و یا عدم استفاده از آن به خود شما واگذار می شود.

انتقال ترافیک HTTP به سوی اسکوئید

به منظور استفاده از اسکوئید در حالت شفاف باید ترافیک موجود در شبکه از طریق مسیریابها و یا سوییچ ها به سوی اسکوئید فرستاده شود که در ادامه به چگونگی انجام این کار می پردازیم.

استفاده از مکانیسم مسیریابی مسیریابها به منظور انتقال ترافیک

درصورتکیه تمامی ترافیک شبکه از یک یا چند مسیریاب به صورت متمرکز عبور می کنند می توان از توانایی های مسیریابها به منظور هدایت ترافیک به سوی پراکسی سرور اسکوئید استفاده کرد بنابراین سیاست های کلی مسیریابهای شبکه باید به گونه ای باشد که تمامی ترافیک موجود در شبکه که از طریق درگاه ۸۰ مبادله می شود به سوی اسکوئید هدایت و مابقی ترافیک موجود که از طریق سایر پروتکلها و درگاهها مبادله می شوند به طور مستقیم به سوی سرورهای مقصد هدایت شوند .شکل کلی این عمل به صورت زیر به تصویر کشیده شده است:

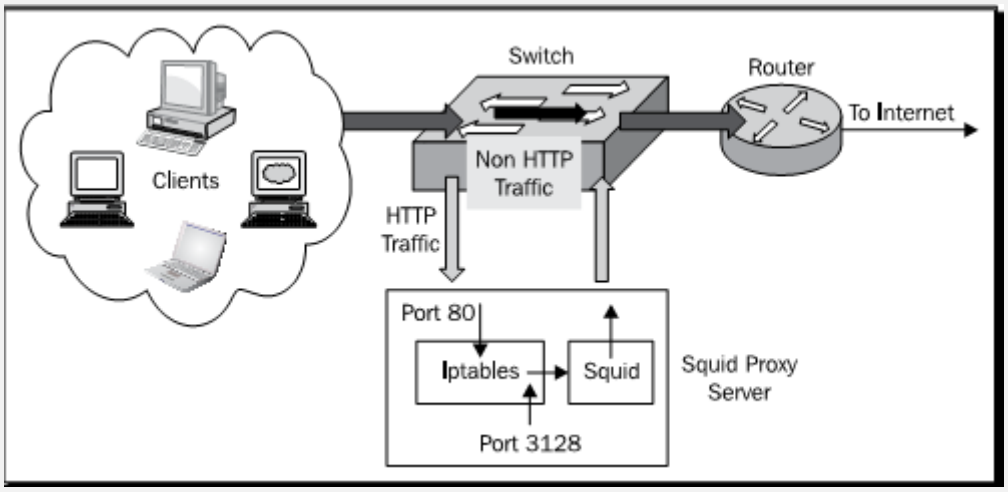


تصویر ۱۰-۱ چگونگی انتقال ترافیک به سوی اسکوئید از طریق یک مسیریاب رانشان می دهد.

مطابق تصویر بالا تمامی ترافیک موجود برروی درگاه ۸۰ که همان درخواستهای HTTP هستند از طریق مسیریاب به سوی اسکوئید هدایت می شود و سایر ترافیک باقی مانده به طور مستقیم به سوی سرورهای خارجی مقصد هدایت می شوند.یک مسیریاب تنها قادر به تغییر آدرس ip بسته های داده است بنابراین باید با استفاده از یک ابزار فیلترینگ بسته های ip مانند iptables و ipfw ،ترافیک موجود برروی درگاه ۸۰ را به سوی درگاهی که اسکوئید به آن گوش می کند منتقل کنیم.

استفاده از سوییچ ها به منظور انتقال درخواستها

با استفاده از یک سوییچ لایه ی 4 یا 7 نیز می توان درخواستهای HTTP را به سوی اسکوئید انتقال داد.تصویر زیر این مطلب را به روشنی بیان می کند:



تصویر ۱۰-۲ چگونگی انتقال ترافیک و درخواستهای کاربران توسط سوییچ به سوی اسکوئید را نشان می دهد.

مطابق این شکل تمامی درخواستهای HTTP موجود برروی درگاه ۸۰ از طریق سوییچ به سوی اسکوئید منتقل می شود و مابقی ترافیک موجود برروی سایر پروتکل ها به طور مستقیم به سوی سرورهای حارجی مقصد هدایت می شود .

به کارگیری اسکوئید به عنوان پل ارتباطی

در این سناریو، اسکوئید مانند یک دروازه ارتباطی (gateway) برای تمامی کاربران شبکه جهت دسترسی به اینترنت عمل می کند بنابراین تمامی بسته هایی که قرار است به سرورهای خارجی مقصد فرستاده شوند ابتدا باید از سروری که اسکوئید برروی آن درحال فعالیت است عبور کند در اینجا ابزارهای مدیریت بسته ها داده تمامی ترافیک HTTP را به سوی اسکوئید هدایت می کند و مانند گذشته سایر ترافیک موجو نیز به طور مستقیم به سوی سرورهای مقصد فرستاده می

شود.با نگاهی به تصویر بالا و مقایسه ی آن با دو تصویر قبل از آن می توان فهمید که اسکوئید در جایگاه یک مسیریاب و یا سوئیچ با قابلیت مسیریابی (Routing) قرار گرفته است قابلیت مسیریابی اسکوئید و سیستم عامل لینوکس و یونیکس از طریق ابزارهای iptables یا ipfw قابل انجام است.درمیان سه روش معرفی شده در مبحث استفاده از کش سرور در حالت شفاف ،روش سوم یعنی استفاده از سرور اسکوئید به عنوان مسیریاب ساده ترین روش جهت به کارگیری اسکوئید در حالت شفاف است که طبیعتا مسائل مربوط به پیکربندی مسیریابها و سوییچ ها که البته برای برخی افراد کاری مشکل و پیچیده به حساب می آید را نخواهد داشت.

آشنایی با WCCP tunnel

Web Cache Coordination Protocol (WCCP) پروتکل اختصاصی شرکت سیسکو جهت استفاده در مسیریابها به منظور هدایت ترافیک به سوی سرورهای پراکسی به صورت بلادرنگ می باشد .می توانیم از پروتکل WCCP در صورت نبود سوییچ های لایه ۴ استفاده کرد.این روش معمولا بر سایر روش های موجود انتقال داده ها نظیر Policy-based Routing که فقط قادر به ارتباط با یک پراکسی سرور در آن واحد است برتری دارد در این روش می توان به طورهمزمان با چند پراکسی سرور ارتباط برقرار کرد.WCCP ویژگیهای بسیاری از جمله قابلیت توازن بار(Load balancing)،مقیاس گذاری(Scaling) ،تحمل خطا(fault tolerance) و پشتیبان پذیری(failover) را دارا می باشد.هنگام استفاده از WCCP ،یک تانل ارتباطی از نوع (Generic Routing Encapsulation) GRE میان مسیریاب و ماشینی که پراکسی سرور برروی آن فعال است برقرار می شود درخواستهای انتقال داده شده از طرف مسیریاب به صورت بسته های GRE کپسوله می شوند و از طریق تانل GRE به سوی پراکسی سرور ارسال می شوند عملیات خارج کردن بسته ها از حالت گرد و انتقال آن به سوی اسکوئید از طریق ماشین میزبانی که حاوی ابزار iptables است انجام می شود سپس اسکوئید درخواستهای دریافتی را از طریق مخزن کش یا سرورهای مقصد پاسخ می دهد و پاسخ مورد نظر را به سوی مسیریاب برمی گرداند و درنهایت مسیریاب نیز پاسخ را به سوی کاربری که آن را درخواست کرده ارسال می کند.جهت کسب اطلاعات بیشتر در مورد پیکربندی دستگاههای مختلف سیسکو و سیستم عاملهای میزبان و همچنین تنظیمات مربوط به اسکوئید از صفحه ی http://wiki.squid-cache.org/Features/Wccp2 بازدید نمائید.

پیاده سازی کش سرور در حالت شفاف

پس از بررسی مزایا و معایب استفاده از کش سرور در حالت شفاف و تصمیم گیری درمورد استفاده از آن باید سه مقوله اصلی مورد توجه قرار گیرد،مورد اول پیکربندی مناسب مسیریابها یا سوییچ ها در شبکه که البته در صورتکیه اسکوئید به صورت دروازه یا پل قرار گیرد این مورد منتفی است،مورد دوم پیکربندی ابزارهای مدیریت بسته ها مانند iptables یا ipfw برروی سروری که اسکوئید برروی آن فعال است و در نهایت پیکربندی اسکوئید است. در ادامه به بررسی این

موارد می پردازیم.

پیکربندی تجهیزات شبکه

در صورتیکه از یک مسیریاب یا سوییچ به منظور انتقال ترافیک به سوی اسکوئید استفاده می کنیم در مرحله ی نخست باید مسیریاب یا سوییچ مورد نظر را منظور شناسایی درخواست های HTTP و انتقال آنها به سوی اسکوئید پیکربندی نمائیم که با توجه به تنوعی که در میان تجهیزات شبکه وجود دارد جهت کسب مهارت لازم برای پیکربندی صحیح آن به مستندات هریک از تجهیرات مربوطه مراجعه کنید.

پیکربندی سیستم عامل

هنگامی که بسته های داده ای که از درگاه ۸۰ به سوی ماشینی که اسکوئید برروی آن فعال است ارسال می شود،باید ابزارهای مدیریت بسته های داده مانند iptables یا ipfw جهت برقراری ارتباط موثر با اسکوئید پیکربندی شوند همچنین باید توجه کرد این ارتباط از طریق درگاهی که در تنظیمات پیکربندی اسکوئید مشخص می شود و ابزارهای مدیریت بسته های داده برقرار می شود بنابراین در صورتیکه درگاه مورد نظر از طریق دیوارآتش مسدود شده است باید فعال شود.یکی از نکاتی که باید قبل از پیکربندی ابزارهای iptables یا ipfw به آن توجه کرد این است که سیستم عامل یونیکس یا لینوکسی که اسکوئید برروی آن فعال است تمامی درخواستهایی که از طرف سرورها خارجی و کاربران به سوی آن ارسال می شود را از طریق کرنل سیستم عامل مسدود می کند و اجازه ی عبور آنها را نخواهد داد دلیل این موضوع آن است که آدرس ip سرورهایی که درخواست را به سوی اسکوئید ارسال کرده اند با هیچ یک از آدرس های ثبت شده بروی کارتهای شبکه همخوانی ندارد.برای حل این مشکل باید قابلیت IP forwarding موجود در هسته ی سیستم عامل فعال شود تا مجوز عبور داده ها صادر شود.

فعال سازی IP forwarding

راههای مختلفی به منظور فعال سازی قابلیت IP forwarding برروی سیستم عاملهای مختلف وجود دارد که در ادامه به تعدادی از آنها اشاره می کنیم.

۱. جهت فعال سازی IP forwarding برروی توزیعهای لینوکسی هر یک از راههای زیر پیشنهاد می شود:

استفاده از دستور sysctl :

```
sysctl -w net.ipv4.ip_forward=1
```

این روش IP forwarding را بدون نیاز به راه اندازی مجدد سیستم عامل فعال می کند اما در راه اندازی بعدی

این قابلیت دوباره غیرفعال می شود و باید دستور بالا دوباره اجرا شود.

اضافه کردن خط زیر به فایل /etc/sysctl.conf :

net.ipv4.ip_forward = 1

پس از اضافه کردن خط بالا در فایل مذکور پس از ذخیره کردن تنظیمات با وارد کردن دستور/etc/sysctl.conf sysctl –p قابلیت IP forwarding فعال شده و در راه اندازی های این قابلیت فعال باقی می ماند و دیگر نیازی به انجام مراحل بالا نیست.

۲. فعال سازی IP forwarding برروی سیستم عامل های BSD نیز تقریبا مشابه تنظیمات توزیعهای لینوکس است:

استفده از دستور sysctl:

sysctl -w net.inet.ip.forwarding=1

در این روش نیازی به راه اندازی مجدد نیست اما توجه داشته باشید که در سیستم عاملهای OpenBsd و DragonFlyBsd نیازی به استفاده از دستور –w نیست.

اضافه کردن خط زیر به فایل /etc/rc.conf :

gateway_enable="YES"

جهت اعمال تنظیمات و فعال سازی IP forwarding یک بار سیستم عامل را راه اندازی مجدد نمائید البته توزیع OpenBsd نیازی به راه اندازی مجدد نیست.

تغییر نشانی بسته ها به سوی اسکوئید

پس از فعال سازی IP forwarding ،باید ابزارهای مدیریت بسته های داده ای جهت انتقال ترافیک به سوی اسکوئید پیکربندی شود ابزار مورد نیاز در این مرحله iptables یا ipfw است که به منظور انتقال ترافیک از درگاه ۸۰ به سوی درگاهی که اسکوئید به آن گوش می کند مانند ۳۱۲۸ به کار می رود.در ادامه به چگونگی انجام این کار می پردازیم:

توجه داشته باشید که در دستورات iptables ی که در ادامه مطرح می شوند آدرس IP اسکوئید ۱۹۲.۰.۲.۲۵ فرض شده است.مراحل انجام این کار به صورت زیر است:

- درتوزیعهای لینوکس:

به منظور هدایت ترافیک موجود برروی درگاه ۸۰ به سمت اسکوئید از دستورات زیر استفاده می کنیم:

```
iptables -t nat -A PREROUTING -s 192.0.2.25 -p tcp --dport 80 -j ACCEPT
iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to-destination 192.0.2.25:3128
iptables -t nat -A POSTROUTING -j MASQUERADE
```

در فهرست دستورات مطرح شده،دستور اول از انتقال ترافیک از اسکوئید به سوی خودش جلوگیری می کند در صورتیکه این دستور به کار نرود،حلقه ایجاد شده ودرخواستها به سوی مقصد هدایت نمی شوند.خط دوم تمامی درخواستهای موجود برروی درگاه ۸۰ را به سوی درگاه و آدرسی که اسکوئید به آن گوش می کند هدایت می کند که درگاه مورد نظر در اینجا ۳۱۲۸ فرض شده است.خط آخر دستورات قابلیت Network Address Translation (NAT) را فعال می کند (اطلاعات بیشتر در این زمینه در http://en.wikipedia.org/wiki/Network_address_translation موجود است).علاوه بر موارد فوق می توان قابلیت tproxy یا پراکسی سرور تمام شفاف را پیاده سازی کرد.البته توجه داشته باشید که این کار مستلزم به کارگیری هسته های جدید لینوکس و نسخه ۴.۱ به بعد iptables است از آنجا که شرح کامل مراحل این کار از حوصله ی این کتاب خارج است جهت کسب اطلاعات بیشتر به نشانی http://wiki.squid-cache.org/Features/Tproxy4 مراجعه نمائید.

- درتوزیع BSD:

ابزارهای مدیریت بسته ی زیادی در سیستم عامل BSD موجود است اما در سیستم عامل OpenBsdابزار Packet Filter یا pf فراهم شده است که برای آگاهی از جزئیات پیکربندی آنها به http://www.openbsd.org/faq/pf همچنین جهت آشناییی با ابزارهای مشابه موجود در سیستم عامل NetBsd به آدرس http://www.netbsd.org/docs/network/pf.html مراجعه کنید.

پیکربندی اسکوئید

تاکنون مراحل و تنظیمات مورد نیاز جهت انتقال ترافیک از درگاه ۸۰ به سوی درگاه ۳۱۲۸ را آموختیم در ادامه تنظیمات پیکربندی اسکوئید را بررسی می کنیم:

پیکربندی درگاه HTTP

آخرین مرحله برای راه اندازی یک کش سرور در حالت شفاف ،اضافه کردن خطوط زیر به فایل پیکربندی اسکوئید است:

```
http_port 3128 intercept
http_port 8080
```

در این تنظیمات درگاه ۳۱۲۸ به منظور انتقال ترافیک به سوی اسکوئید به کار میرود و درگاه ۸۰ نیز برای هدایت سایر ترافیک شبکه به سوی سرورهای مقصدبه کار می رود البته استفاده از درگاه ۸۰۸۰ اختیاری بوده اما استفاده از آن در مواردی چون مدیریت دسترسی پراکسی سرور و سایر درخواستها که با حالت شفاف سازگاری ندارند به کار می رود.

پس از طی مراحل بالا هم اکنون اسکوئید قادر به دریافت و ارسال ترافیک شبکه در حالت شفاف می باشد.

خلاصه فصل

دراین فصل ویژگی های کش سرور درحالت شفاف و جزئیات مربوط به آن معرفی شدند همچنین چگونگی انتقال ترافیک از سایر دستگاههای شبکه به سوی اسکوئید بیان شد.مباحث مطرح شده در این فصل شامل موارد زیر است:

- معرفی کش سرور در حالت شفاف و بیان چگونگی عملکرد آن
- راههای مختلف چگونگی برپاسازی کش سرور در حالت شفاف
- مزایا و معایب کش سرور شفاف
- پیکربندی سیستم عامل جهت انجام عمل IP forwarding
- پیکربندی ابزارهای مدیریت بسته های داده جهت انتقال ترافیک به سوی اسکوئید
- راههای مختلف برپاسازی کش سرور شفاف برروی سیستم عاملهای مختلف

فصل یازدهم

ابزارهای بازنویسی و تغییر نشانی آدرسهای URL

در این فصل به چگونگی به کارگیری ابزارهای تغییر نشانی و بازنویسی آدرسهای URLدر اسکوئید آشنا خواهیم شد.این ابزارها و دستورات به منظور اعمال کنترل بیشتر برآدرس های قابل دسترس برای کاربران ایجاد دشه است هر چند در برخی موارد می توان از آنها در کارهای تبلیغی نیز استفاده کرد.

به طو رکلی مطالبی که در این فصل مطرح میشوند شامل موارد زیر است:

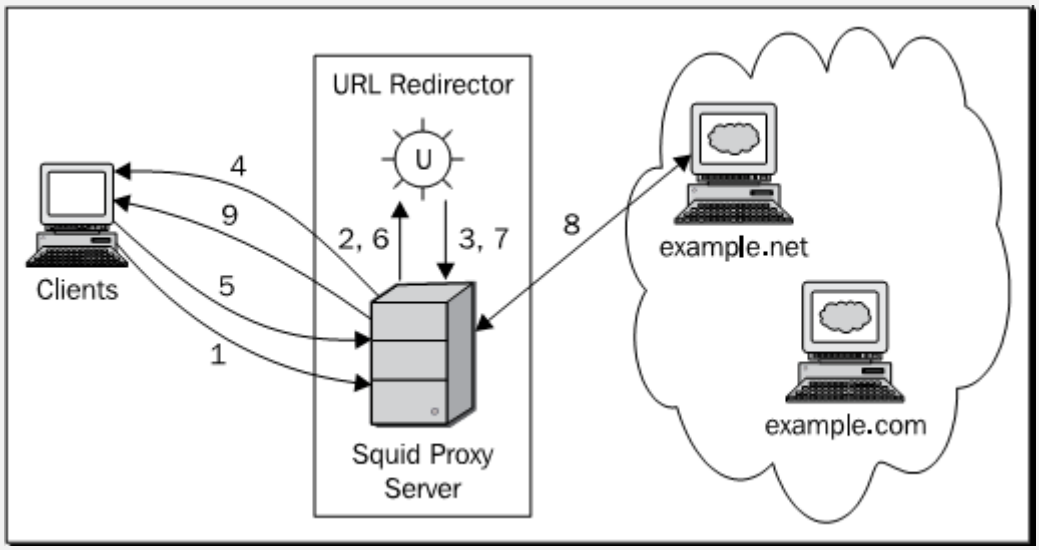
- آشنایی با ابزارهای تغییر نشانی و بازنویسی آدرسهای URL
- نوشتن و ایجاد ابزار مورد نظر خود
- پیکربندی اسکوئید
- آشنایی با ابزارهای بازنویسی محبوب

بازنویسی و تغییر نشانی آدرسهای URL

ابزارهای تغییر نشانی URLها ، به ابزارهایی اطلاق می شوند که از طریق یک پردازش خارجی و مجزا از اسکوئید قادر به تغییر نشانی آدرسها از مسیر اصلی و هدایت آنها به سوی آدرسهای مورد نظر ما است.به طور مشابه ابزارهای بازنویسی URLنیز به ابزارهایی گفته می شوند که در کنار ابزارهای تغییر نشانی قادر به بازنویسی آدرسها با آدرسهای جدید و در نهایت هدایت آن به سوی کاربران است در واقع هنگامی که یک آدرس URL باز نویسی می شود ،نشانی جدید به صورت کاملا شفاف و نامرئی توسط کاربر قابل دریافت می شود و کاربر به جای مشاهده محتوای اصلی در نشانی مورد نظر ، نشانی و محتوای جدید را دریافت و مشاهده می کند.ابزارهای تغییر نشانی آدرسها قادر به ارسال پیام های تغییر نشانی پروتکل HTTP نظیر ۳۰۱، ۳۰۷،۳۰۳،۳۰۲ و سایر کدهای مشابه همراه با یک آدرس مستقیم به سوی کاربران HTTPاست بنابراین هنگامی که یک کاربر یک کد درخواست تغییر نشانی را همراه با نشانی مورد نظر دریافت می کند ، همان کاربر نشانی جدید را به جای نشانی قدیمی درخواست می کند.درواقع مهمترین تفاوت عمده ی میان ابزارهای بازنویسی و تغییر نشانی نشانی های URLاین است که در تغییر نشانی نشانی ها کاربر بادریافتهای کدهای تغیر مسیر از فرایند انجام کار آگاه می شود درحالیکه در ابزارهای بارنویسی نشانی های URL ، کاربر بدون اطلاع از فرایند در حال وقوع ، نشانی و محتوای جدید را دریافت می کند.درادامه به بررسی دقیقتر چگونگی عملکرد این ابزارها می پردازیم.

آشنایی با ابزارهای تغییر نشانی آدرس های URL

با دقت در شماره گذاری موجود در تصویر ۱۲-۱ می خواهیم طرز عملکرد یک ابزار تغییر نشانی نشانی را همراه با اسکوئیدو پردازش های مورد نیاز برای تغییر نشانی یک نشانی هنگامیکه یک کاربر آدرس http://example.com/index.html را درخواست می کند را بیاموزیم:



تصویر ۱۲-۱ چگونگی عملکرد یک ابزار تغییر تغییر نشانی نشانی ها را همراه با اسکوئید به تصویر می کشد.

توجه داشته باشید مراحل نه گانه ی زیر با توجه به شماره ی های موجود برروی تصویر ۱۲-۱ شرح داده می شوند:

۱. کاربر نشانی http://example.com/index.html را درخواست می کند.

۲. پراکسی سرور اسکوئید درخواست مورد نظر را دریافت و جزئیات لازم را به سوی ابزار تغییر نشانی مورد نظر ارسال می کند.

۳. ابزار تغییر نشانی URL، اطلاعات لازم را دریافت و پردازش می کند و با دریافت کد ۳۰۳ مبنی بر تغییر نشانی نشانی مورد نظر به اسکوئید اطلاع می دهد که نشانی مورد نظر باید با یک نشانی جدید تعویض شود.به بیان ساده تر در این مرحله اسکوئید از تغییر نشانی نشانی ارسال شده مطلع می شود.

۴. اسکوئید با توجه به اطلاعات دریافت شده از ابزار تغییر نشانی ، پیام تغییر نشانی را همراه با نشانی اصلی به سوی کاربر ارسال می کند.

۵. کاربر با دریافت پیام ارسال شده از اسکوئید ، نشانی اصلی را همراه با نشانی ای که باید با آن جابجا شود یعنی http://example.net/index.html را دریافت می کند.

۶. هنگامی که اسکوئید درخواست جدیدی را دریافت کند ، درخواست مورد نظر با دوباره به سوی ابزار تغییر نشانی نشانی ها ارسال می کند.

۷. ابزار تغییر نشانی با دریافت نشانی ارسال شده به اسکوئید اطلاع می دهد که درخواست و نشانی مورد نظر نیازی به پردازش و تغییر نشانی نداشته و اسکوئید به طور مستقیم می تواند درخواست کاربر را پاسخ گوید.

۸. اسکوئید نشانی http://example.net/index.html را برای کاربر درخواست کننده نمایش می دهد.

۹. پاسخ دریافت شده از سرور اصلی وب سایت example.net به کاربر درخواست کننده تحویل داده می شود.

با دقت در تصویر بالا و مراحل شرح داده شده در آن ، می توان فرایند منطقی پردازش یک نشانی را از مرحله ی درخواست از سوی کاربر تا مراحل پردازش درخواست و تحویل دوباره به کار بر را مشاهده و درک کرد.درادامه می خواهیم با مهمترین کدهای مورد استفاده در پروتکل HTTP به منظور استفاده در فرایند تغییر نشانی و موارد کاربرد آنها آشنا شویم.

کدهای وضعیت HTTP مورد استفاده در تغییر نشانی URLها

در بخش قبل دریافتیم که با به کارگیری کدهای نشان دهنده وضعیت تغییر نشانی درخواست های HTTPمی توان کاربران را به سوی آدرسهای جدید هدایت کرد.حال می خواهیم با موارد به کارگیری این کدها آشنا شویم.

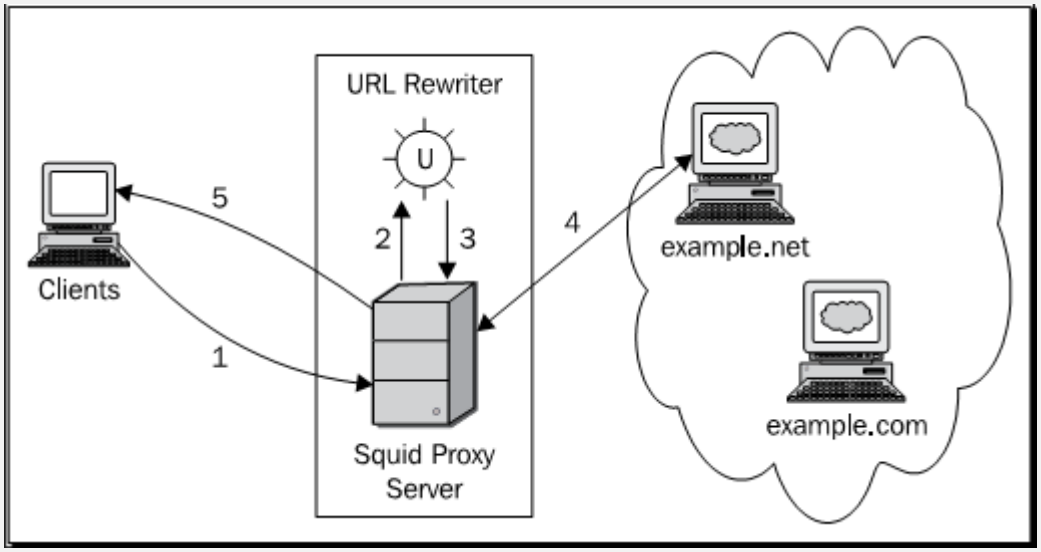
شماره ی کد	توضیحات و موارد کاربرد
۳۰۱	کد ۳۰۱ بیان می کند که درخواست ارسال شده از سوی کاربر به طور دائم انتقال داده شده و سایر درخواستهای بعدی نیز باید به این نشانی تغییر یابند (تغییر نشانی صورت می گیرد). این کد فقط هنگامی که اسکوئید در حالت پراکسی معکوس قرار داشته باشد قابل استفاده خواهد بود.
۳۰۲	کد وضعیت ۳۰۲ نشان می دهد که درخواست ارائه شده می تواند به طور مستقیم و بدون هیچ تغییر نشانی ای پاسخ داده شود.کاربر د این کد عمدتا برای درخواستهای از نوع GETو HEADبه کار می رود.
۳۰۳	این کد بدین معنی است که درخواست مورد نظر به طور مستقیم قابل پاسخگویی است اما درخواست مورد نظر باید از نوع GETباشد.این کد به همراه درخواستهای POSTو PUTقابل استفاده خواهد بود.
۳۰۵	کد ۳۰۵ بدین معنی است که کاربر مورد نظر باید از یک پراکسی سرور جهت دریافت داده های مورد نیاز خود استفاده کند.این کد جهت استفاده همراه با پراکسی سرورهای حالت شفاف که نیاز به تغییر حالت به یک پراکسی ارجاع دهنده جهت پاسخگویی به درخواست کاربران دارند به کار می رود.
۳۰۷	کد ۳۰۷ به معنای تغییر نشانی URLبه یک نشانی جدید است اما تمامی ویژگیهای نشانی قدیمی نیز باید به نشانی جدید منتقل شود به عبارت دیگر تمامی متدهای موجود در درخواست ونشانی قدیمی باید به نشانی جدید انتقال داده شود.به طور کلی این کد به همراه درخواستهای CONNECT/HTTPS به کار می رود.

کدهایی که در جدول بالا معرفی شدند از جمله پرکاربردترین کدهای موجود هستند.جهت کسب اطلاعات بیشتر و آشنایی با فهرست کدهای وضعیت به http://en.wikipedia.org/wiki/List_of_HTTP_status_codes#3xx_Redirection مراجعه نمائید.

آشنایی با ابزارهای بازنویسی نشانی های URL

به طور کلی مکانیسم کاری ابزارهای بازنویسی نشانی های URLهمانند ابزارهای تغییر نشانی است.تفاوت عمده ی این دو در این است که در ابزارهای بازنویسی کاربر از تغییر نشانی صورت گرفته آگاه نبوده و این کار به صورت کاملا مخفیانه انجام می شود.فرض کنیم کاربری نشانی http://example.com/index.html را در مرورگر خود وارد کرده و می خواهد محتوای موجود در این وب سایت را مشاهده نماید درحالیکه در همین شبکه یک ابزار بازنویسی به همراه اسکوئید مطابق تصویر

زیر وجود دارد:



تصویر ۱۲-۲ موقعیت و عملکرد یک ابزار بازنویسی و اسکوئید در شبکه را نشان می دهد.

شماره های موجود در تصویر بالا مراحل پردازش یک درخواست را میان ابزاربازنویسی و اسکوئید را بیان می کند.فرایندی که در هر مرحله صورت می گیرد به صورت زیر است:

۱. کاربر مورد نظر نشانی http://example.com/index.html را از پراکسی سرور ما درخواست می کند.

۲. اسکوئید درخواست مورد نظر را دریافت کرده و جزئیات آن را برای ابزار بازنویسی ارسال می کند.

۳. ابزاربازنویسی جزئیات دریافت شده از اسکوئید را پردازش کرده و به اسکوئید اطلاع می دهد که نشانی مورد نظر باید با نشانی http://example.net/index.html تغییر یابد و یا به عبارت ساده تر نشانی قدیمی باید با نشانی جدید بازنویسی شود.

۴. اسکوئید نشانی جدید یعنی http://example.net/index.html را دریافت کرده و در این مرحله به جای برقراری ارتباط با سرور example.comبا سرور نشانی جدید یعنی example.netارتباط برقرار می کند.

۵. و درنهایت اسکوئید پاسخ درخواست ارسال شده به سوی سرور example.netرا دریافت و آن را به کاربر درخواست کننده تحویل می دهد.

حال که با طرز کار یک ابزار بازنویسی نشانی آشنا شدید باید به این نکته ی مهم و جالب توجه کنید که کاربر نهایی

دریافت کننده در مثال بالا احساس م یکند که پاسخ درخوسات خود را از سرورهای example.comو نه example.netدریافت کرده است و درواقع مهمترین مزیت این ابزارها مخفی بودن تمامی مراحل پردازش اطلاعات از دید کاربران است.

مسائل مربوط به ابزارهای بازنویسی نشانی های URL

درادامه می خواهیم با برخی از مشکلات و مسائلی که در هنگام به کارگیری ابزارهای بازنویسی ممکن است اتفاق بیفتد و منجر به بروز نتایج غیرقابل پیش بینی شود آشنا شویم:

- بازنویسی برخی از نشانی های URLممکن است باعث به وجود آمدن پاسخ های غیرقابل پیش بینی از طرف پراکسی سرور شود به علاوه درخواست مورد نظر ممکن است قبلا توسط کاربران دیگر درخواست شده باشد و جزء داده های کش شده محسوب شود که این موضوع در برخی موارد ممکن است موجب کش شدن داده ها و نشانی هایی که حاوی کدهای مخرب باشند شود البته به طور کلی این یک مشکل محسوب نمی شود زیرا در صورتکیه درخوساتها از قبل توسط کاربران درخوسات شده باشد ، مانند سایر درخوساتها اسکوئید از مخزن کش جهت پاسخگویی به آنها استفاده می کند.

- بازنویسی upload، درخواستهای نوع POSTو WebDAVممکن است موجب ایجاد تغییرات غیرقابل پیش بینی برروی سرورهای اصلی و مقصد شوند.

- درصورتیکه یک ابزار بازنویسی یک نشانی غیرمعتبر را به اسکوئید برگرداند این امر ممکن است موجب ایجاد یک رفتار غیرقابل پیش بینی از سوی اسکوئید شود.به عنوان مثال ممکن است مرورگر مورد استفاده ی شما کاراکتر (#) را به عنوان یک کاراکتر درنظر بگیرد اما در صورتکیه یک ابزار بازنویسی یک نشانی URLکه حاوی کاراکتر (#) است را به اسکوئید برمی گردند اسکوئید این کاراکتر را تشخیص نمی دهد و نمی تواند با این کاراکتر به طور صحیح رفتار کند از این رو اسکوئید نشانی مورد نظر را نامعتبر تشخیص داده و پیغام خطایی را به کاربر درخواست کننده ارسال می کند و یا درخواست را در حالت bypassقرار می دهد.درحالیکه یک ابزار تغییر مسیر با توجه به سطح توانایی مرورگر کاربر نحوه برخورد با درخواستهایی که در آنها کاراکتر (#) به کاررفته شده باشد را تشخیص می دهد.

- بازنویسی درخواستهای CONNECT/HTTPS ممکن است باعث ایجاد خطاهای مربوط به پروتکل HTTPSبرروی کانالهای امنیتی شود.

همانطور که می بینیم ابزارهای بازنویسی نشانی های URLدر مقایسه با ابزارهای تغییر مسیر دارای مشکلات

بیشتری هستند به علاوه ابزارهای تغییر مسیر نشانی های URLکاربر را از انجام عمل تغییر مسیر نشانی ها آگاه می کند.

توانایی ابزارهای تغییر مسیر در آگاه کردن کاربران از ایجاد تغییرات انجام شده مزیت بزرگی برای توسعه دهندگان وب محسوب می شود زیرا با استفاده از این قابلیت می توان کاربران را به صفحات خطا و یا یک متن هدایت کرد و یا تبلیغات ناخواسته ی موجود در صفحات وب را مسدود کرد و همچنین کاربران را به محتوای موجود در شبکه ی محلی خود در صورت نیاز هدایت کرد.

اسکوئید و ابزارهای بازنویسی و تغییر مسیر نشانی های URL

اسکوئید و ابزارهای تغییر مسیر و یا بازنویسی ارتباط تگاتنگی دارند به طوریکه درخواستهای دریافت شده توسط اسکوئید جهت پردازش و احتمالا انجام تغییرات نظیر تغییر مسیر و یا بازنویسی به سوی این ابزارها ارسال می شوند.در ادامه می خواهیم با چگونگی انجام این عمل بیشتر آشنا و از جزئیات این ارتباطات بیشتر آگاه شویم.

رابط ارتباطات

ابزارهای تغییر مسیر نشانی و بازنویسی URLها از رابط ارتباطی بسیار ساده ای استفاده می کنند که یادگیری آن به آسانی پیاده سازی آنهاست.به ازای هردرخواستی که اسکوئید به سوی این ابزارها ارسال می کند ، جزئیات زیر را به همراه دارد:

ID URL client_IP/FQDN username method myip=IP myport=PORT [kv-pairs]

جدول زیر هر یک از فیلد های اطلاعاتی بالا را معرفی و خصوصیات آن را بیان می کند:

توضیحات	فیلد مورد نظر
هر IDبه منظور شناسایی یک درخوسات توسط اسکوئید و ارسال آن به سوی ابزار تغییر نشانی به کار می رود.این شماره شناسایی بهمظنور مشخص کردن درخواستهای مختلف و جلوگیری از عوض شدن نشانی ها به کار می رود.یکی دیکر از دلایل استفاده از این نام ، پاسخگویی به طیف وسیعی از درخواستها به طور همزمان است.	ID
فیلد URLتعیین کننده ی نشانی درخواست کاربران است که به سوی ابزار بازنویسی ارسال می شود و دوباره به کاربر بازگردانده می شود.	URL
این فیلد نشانی IPکاربران را مشخص می کند.	Client_IP
FQDN یا (Fully Qualified Domain Name) نام کامل و دقیق دامنه از سوی کاربر را تعیین می کن.درصورتکیه فیلد تعیین نشده باشد ، به جای آن یک خط فاصله قرار می	FQDN

	گیرد.توجه داشته باشید که FQDN تنها زمانی قابل استفاده است که جستجوی معکوس DNS برای نشانی های IPوجودداشته باشد و اسکوئید قادر به تبدیل نامهای دامنه به نشانی های IPو بالعکس باشد.
Username	فیلد username نام کاربری مربوط به کاربری را که درخواستی را ارسال کرده در صورت وجود مشخص می کند.فیلد نام کاربری درصورتیکه موجود نباشد به جای آن یک خط فاصله به عنوان مقدار موجود در این فیلد قرار می گیرد.
Method	این فیلد متدهای مربوط به پروتکل HTTP برای درخواستهای ارسال شده از سوی کاربران را مشخص می کند که مقادیر آن شامل GET، POST ، DELETEو ... می باشد.
Myip=IP	myip=IP (myip=IP_ADDRESS) نشانی IPدریافت شده توسط اسکوئید از سوی کاربران ارسال کننده ی درخواست را مشخص می کند.این گزینه تنها زمانی مفید است که اسکوئید بیش از یک کارت شبکه داشته باشد و همچنین نشانی های قابل دسترس اسکوئید بیش از یک نشانی تعریف شده باشد.
Myport=PORT	فیلد(myport=PORT_NUMBER) myport=PORT شماره ی درگاهی را که اسکوئید به درخواستهای ارسال دشه توسط کاربران گوش می کند و از طریق آن درخواستها را دریافت می کند را مشخص می کند.این فیلد نیز تنها زمانی مفید است که برای اسکوئید بیش از یک درگاه تعریف شده باشد.
Kv-pairs	این فیلد سایر مقادیر موجود برای کلیدها را جهت دسترسی ابزارهای بازنویسی به منظور استفاده در آینده را مشخص می کند.

با توجه به توضیحاتی که در جدول بالا ارائه شد ، اسکوئید و سایر ابزارهای تغییر مسیر نشانی و بازنویسی URLها به طور مداوم در حال تبادل اطلاعات و پیام ها هستند در واقع اسکوئید از این ابزارها به عنوان پردازش خارجی به منظور اعمال تغییرات بر نشانی ها استفاده می کند.خطوط زیر نمونه ای پیام های مبادله شده میان اسکوئید و این ابزارها را نشان می دهند:

http://www.example.com/ 127.0.0.1/localhost – GET myip=127.0.0.1 myport=3128

http://www.example.net/index.php?test=123 192.0.2.25/- john GET

myip=192.0.2.25 myport=8080

http://www.example.org/login.php 198.51.100.86/- saini POST

myip=192.0.2.25 myport=8080

همانطور که مشاهده می کنید تمامی فیلدهای شرح داده شده در جدول بالا دراین خطوط مشاهده می شود از جمله

نشانی وب سایت ، نشانی IPکاربر ، درگاهی که اسکوئید به درخواستها گوش می کند و....

نکته: به دلیل تبادل اطلاعات شخصی و حساس کاربری میان اسکوئید و این ابزارها از معتبر بودن ابزارهای مورد استفاده اطمینان حاصل نمائید زیرا در صورت بروز مشکلات امنیتی اطلاعات کاربرانی که توسط ابزارهای تغییر نشانی یا بازنویسی مورد پردازش قرار می گیرند با خطر جدی سوءاستفاده مواجه می شوند.

ابزارهای تغییر نشانی همواره به خواندن و پردازش داده های ارسال شده از سوی اسکوئید مشغولند و تنها زمانی یک پیام و درخواست را تمام شده فرض می کنند که با عبارت EOF (end of file)در پایان آنها برخورد کنند در غیر اینصورت حلقه های به وجود می آیند که در این حالت اسکوئید در تلاش برای اجرای تعداد بیشتری از ابزارها برمی آید ، در این حالت پیا م های هشداری مبنی بر وجود مشکل در اجرای ابزارهای تغییر مسیر نشانی به صورت زیر در فایل cache.log نمایان می شوند:

WARNING: redirector #1 (FD 8) exited |۲۲:۰۱:۱۹ ۰۸/۱۱/۲۰۱۰

نوشتن یک ابزار تغییر مسیر نشانی

با استفاده از مفاهیمی که تاکنون در مورد ابزارهای تغییر مسیر نشانی و بازنویسی آموخته ایم می توانیم این ابزارها از طریق کدنویسی ایجاد و سفارشی سازی کرد.مثال زیر نمونه ای از کدهای مربوط به یک ابزار تغییر نشانی که با زبان پایتون نوشته شده است را نشان می دهد:

```
/!#usr/bin/env python

import sys

def redirect_url(line, concurrent:(

 list = line.split(' ')

۱ # st or 2nd element of the list

# is the URL depending on concurrency

 if concurrent:

  old_url = list[1]

 else:
```

```
  old_url = list[0]
#Do remember that the new_url
# should contain a '\n' at the end.
 New_url = '\n'
# Take the decision and modify the url if needed
 if old_url.endswith('.avi'):
#   Rewrite example
  new_url = 'http://example.com/' + new_url
 elif old_url.endswith('.exe'):
#   Redirect example
  new_url = '302:http://google.co.in/' + new_url
 return new_url
def main(concurrent = True):
# the format of the line read from stdin with concurrency is
# ID URL ip-address/fqdn ident method myip=ip myport=port
# and with concurrency disabled is
# URL ip-address/fqdn ident method myip=ip myport=port
 line = sys.stdin.readline().strip()
#   We are to keep doing this unless there is EOF
 while line:
#   new_url will be a URL, URL prefixed with 3xx code
#   or just a blank line.
  New_url = redirect_url(line, concurrent)
```

```
id" =
# Add ID for concurrency if concurrency is enabled.
  If concurrent:
   id += line.split(‘ ‘)[0]' ' +
  new_url = id + new_url
# Write the new_url to standard output and
# flush immediately so that it’s available to Squid
  sys.stdout.write(new_url)
  sys.stdout.flush()
# Read the next line
  line = sys.stdin.readline().strip()
  if __name__ == ‘__main:'__
# Check if concurrency is enabled or not
 main(len(sys.argv) > 1 and sys.argv[1] == ‘-c')
```

این برنامه ابتدا داده های مورد نیاز را از ورودی خوانده و سپس کاراکتر اضافی را که بدان نیاز نیست را حذف می کند.پس از فراخوانی داده های مورد نیاز تابع redirect_url فراخوانی می شود که وظیفه ی آن مرتب کردن داده های ورودی و تقسیم آن برروی فضاهای خالی است پس از آن نشانی هایی که به پسوند avi. ختم شوند به یک صفحه ی خطای محدودیت دسترسی که قبلا طراحی شده هدایت می شود همچنین نشانی هایی که به پسوند exe. ختم شوند به سوی صفحه ی ای که حاوی پیام هشداری مبنی احتمال آلودگی به ویروس هدایت می شوند.

پیکربندی اسکوئید

پس از اینکه ابزار تغییر و یا بازنویسی نشانی های URLرا طراحی و ایجاد کردیم زمان آن است که اسکوئید را جهت برقراری ارتباط با این ابزارها پیکربندی نمائیم.بدین منظور دستوراتی ایجاد شده اند که با استفاده از آنها می توان

چگونگی استفاده ی اسکوئید از این ابزارها را مدیریت کند.در ادامه با این دستورات آشنا می شویم.

مشخص کردن مسیر فایلهای اجرایی ابزار تغییر/بازنویسی نشانی های URL

با استفاده از دستور url_rewrite_program میتوان مسیر و نام ابزار مورد نظر خود را به اسکوئید معرفی کرد.همچنین از طریق این دستور می توان سایر دستورات و آرگومانهای مورد نیاز این ابزارها را در مقابل آن اضافه نمود.خطوط زیر نمونه ای تنظیمات پیکربندی اسکوئید را نشان می دهند:

```
url_rewrite_program /opt/squid/libexec/custom_rewriter
url_rewrite_program /usr/bin/python /opt/squid/libexec/my_rewriter.py
url_rewrite_program /usr/bin/python /opt/squid/libexec/another_
rewriter.py –concurrent
```

نکته: اسکوئید فقط قادر به استفاده از یک ابزار تغییر مسیر و یا بازنویسی نشانی در آن واحد است بنابراین ما فقط مجاز به استفاده از یکی خطوط **url_rewrite_program** *در تنظیمات پیکربندی اسکوئید هستیم.*

مدیریت تعداد موارد اجرای ابزار(رشته های پردازشی) تغییر مسیر نشانی

پس از تعیین مسیر فایلهای اجرایی ابزارهای تغییر مسیر نشانی های URL ، به منظور بالا بردن سطح عملکرد و توانایی پردازشی این ابزارها با استفاده از دستورurl_rewrite _children می توان تعداد رشته های پردازشی از ابزار را که در آن واحد قادر به اجرا هتند را تعیین کرد در واقع اگر ابزار را یک ماهی فرض کنیم رشته های پردازشی فرزندان آن عمل می کنند و قادرند درخواستهای متعددی را به طور همزمان پاسخ دهند.ساختار کلی url_rewrite_children به صورت زیر است:

```
url_rewrite_children CHILDREN startup=N idle=N concurrency=N
```

در این خط تنظیمات ، پارامتر CHILDREN حداکثر تعداد موارد فرزندان ابزار را تعیین می کند که اسکوئید اجازه ی اجرای آنها به طور همزمان را صادر می کند.همواره در انتخاب مقدار این پارامتر دقت داشته باشید که اختصاص مقدار خیلی کم به آن موجب ایجاد تاخیر در ارسال پاسخها به طور همزمان به سوی اسکوئید شده شبکه را با تاخیر مواجه می کند و همچنین اختصاص مقدار بسیار زیاد نیز باعث مصرف بیش از اندازه ی حافظه ی رم و اعمال بار پردازشی زیاد به پردازنده ی سرور می شود که در نتیجه رعایت نکردن هردو مورد باعث ایجاد اختلال در عملکرد اسکوئید می شود.مقدار پیش فرضی که برای CHILDREN تعیین شده است ۲۰ می باشد.پارامتر (startup startup=N) کمترین تعداد رشته های پردازشی (فرزندان) به طور همزمان را زمانی که اسکوئید تازه شروع به کار کرده و یا راه

اندازی مجدد شده است را تعیین می کند.در صورتکیه مقدار startup صفر تعیین شود ، اولین رشته پردازشی درحال اجرا به اولین درخواست ارسال شده به سوی آن اختصاص پیدا می کند و مقدار پیش فرض آن نیز صفر تعیین شده است.

نکته: اختصاص مقادیر بسیار کم به پارامتر **startup** *موجب ایجاد تاخیر اولیه هنگام ارسال تعداد زیاد درخواست به سوی ابزار تغییر مسیر نشانی میشود بنابراین سعی کنید با توجه به میزان ترافیک شبکه ی خود مقدار مناسبی را برای آن تعیین کنید.*

پارامتر idle (idle=N)تعداد رشته های پردازشی ای را که همواره باید در حال اجرا اما بدون بار پردازش باشند را تعیین می کند.نحوه عملکرد این رشته ها بدین صورت است که همواره کمترین مقدار تعیین شده همراه با اجرای اسکوئید اجرا شده و سپس با افزایش میزان ترافیک ، بر تعداد آنها افزوده شده و تا آخرین مقدار تعیین شده برای آنها گسترش می یابند.مقدار پیش گزیده ی idle نیز یک است.

پارامتر concurrency (concurrency=N) نیز تعداد درخواستهای همزمانی که نرم افزار تغییر مسیر نشانی باید پردازش کند را تعیین می کندومقدار پیش فرض این پارامتر صفر است که معنای آن پردازش یک درخواست در ازای هر رشته ی پردازشی در آن واحد است.

مدیریت درخواستهای ارسال شده به سوی ابزار تغییر مسیر نشانیURL

به طور پیش فرض ، تمامی درخواستهای دریافت شده توسط اسکوئید به سوی ابزارهای تغییر مسیر نشانی ارسال می شوند درحالیکه به طور کلی چنین رفتاری معقول به نظر نمی رسد. با استفاده از دستور url_rewrite_access میتوان درخوساتهایی را که به سوی ابزارهای تغییر مسیر نشانی ها ارسال می شوند مدیریت نمود.ساختار و قالب کلی url_rewrite_access نیز مانند http_access است و هر دوی آنها به نوعی برای برای اعمال کنترل مستقیم بر محتوا به کار می روند .فرض کنیم می خواهیم درخواستهایی که از دامنه example.com دریافت می شوند را به سوی این ابزار ارسال کنیم.بدین منظور خطوط زیر را به تنظیمات پیکربندی اسکوئید اضافه می کنیم:

```
acl rewrite_domain dstdom .example.com
url_rewrite_access allow rewrite_domain
url_rewrite_access deny all
```

مطابق این خطوط اسکوئید فقط درخواستهایی را که از دامنه example.com و یا هر زیردامنه آن دریافت شده باشد را به سوی این ابزارها ارسال می کند.همانطور که مشاهده می کنید با ترکیب Aclها و دستور url_rewrite_access می توان بر فرایند ارسال درخواستها به خوبی مدیریت نمود.

*نکته: توجه داشته باشید که درخواستهای نوع **POST**و**CONNECT** به هیچ عنوان نباید تغییر مسیر و یا بازنویسی شوند و هر گونه تغییر در این درخواستها موجب ایجاد خطاهای غیر قابل پیش بینی می شود.ایده ی مناسب برای جلوگیری از بروز این مشکل بلوکه کردنشانها از طریق دستور **url_rewrite_access** است.*

Bypass ابزارهای تغییر مسیر در شرایط سخت کاری

زمانی که یک ابزار تغییر نشانی تحت فشار شدید کاری و دریافت درخواستهایی بیش از توان پردازشی خود باشد در این حالت اسکوئید باید منتظر بماند تا این ابزارها نشانی های URLارسال دشه به سوی آنها را پردازش کرده و به سوی اسکوئید ارسال کنند این فرایند موجب ایجاد وقفه و تاخیر در پاسخگویی به درخواستها ی کاربران می شود.در این شرایط با ب کارگیری دستور url_rewrite_bypass می توان انتقال درخواستها به سوی این ابزارها را متوقف کرده و اسکوئید خود شرایط را مدیریت کند.برای این کار کافی است خط زیر را به فایل پیکربندی اسکوئید اضافه کنیم:

```
url_rewrite_bypass on
```

به طور پیش فرض اسکوئید هیچ درخواستی را bypassنمی کند ودر شرایطی که ترافیک شبکه بالاست منتظر می ماند تا ابزار مورد نظر از نظر سطح پردازش به شرایط عادی بازگشته و سپس درخواستهای ارسال شده را پردازش و به اسکوئید بازگرداند.

بازنویسی هدر HOST

هنگامی که از یک ابزار تغییر مسیر نشانی های URLبه منظور ارسال نشانی های تغییر یاقته به سوی اسکوئید و سپس کاربران استفاده می کنیم ، اسکوئید هدر HOST موجود در درخواستهای HTTP را در درخواستهایی که نشانی های اصلی آنها تغییر مسیر یافته بازنویسی می کند.این ویژگی ممکن است درحالتی که اسکوئید در حالت ارجاع دهنده (forward proxy) بدون هیچگونه مشکلی عمل می کند اما به کارگیری آن درمواردی که اسکوئید در حالت پراکسی معکوس قرار دارد ممکن است باعث ایجاد مشکلات مختلفی شود.برای جلوگیری از بروز چنین مشکلاتی می توان مقدار دستور url_rewrite_host_header را به offتغییر داده تا از بازنویسی هدر های hostجلوگیری شود.

نکته: مقدار پیش فرض اسکوئید برای دستور **url_rewrite_host_header** *، بازنویسی تمامی هدر های نوع* **host** *است.*

آشنایی با دستور deny_info

deny_info دستوری است که در فایل پیکربندی اسکوئید برای موارد زیر به کار می رود:

- هدایت کاربران به صفحات خطای از قبل طراحی شده.
- هدایت درخواستهای کاربران به سوی نشانی های از قبل تعریف شده و نمایش اطلاعاتی در مورد علت محدود بودن دسترسی و یا عدم نمایش محتوای درخواستی توسط کاربران از طریق ارسال پیام HTTP 302.
- قطع کردن ارتباطات نوع TCP و از سرگیری آنها.

برای به کارگیری دستور deny_info سه ساختار و قالب کلی به صورت زیر وجود دارد:

```
deny_info CUSTOM_ERROR_PAGE ACL_NAME
deny_info ALTERNATE_URL ACL_NAME
deny_info TCP_RESET ACL_NAME
```

ساختار نحوی مطرح شده در دستورات بالا ، قالب کلی این دستور و نحوه به کارگیری آن به شکل های گوناگون را نشان می دهند.در خط اول پارامتر CUSTOM_ERROR_PAGE یک صفحه ی خطای از قبل طراحی شده به زبان HTMLو یا یک صفحه ی متنی را مشخص می کند که به عنوان جایگزین صفحه ی پیش فرض عدم دسترسی اسکوئید برای کاربران نمایش داده می شود.صفحه ی خطای طراحی شده برای زبان انگلیسی که زبان پیش فرض خطاهای اسکوئید است باید در مسیر ${ prefix}/share/errors/en-us قرار گیرد همچنین صفحات طراحی شده برای سایر زبانها نیز باید در پوشه ی مخصوص هر زبان قرار گیرد.در صورتکیه مایل نیستید صفحات طراحی شده را در مسیر دایرکتوریهای پیش فرض اسکوئید قرار دهید می توانید مسیر دلخواه خود را مثلا به صورت /etc/squid/local-errors/ در این دستور تعریف کنید.

در خط دوم از دستورات درخواستهای کاربر به یک نشانی URL ی که قبلا طراحی شده از طریق دستور ALTERNATE_URL هدایت می شود.در خط آخر نیز TCP_RESET اتصالات برقرار شده توسط کاربر به نوعی قطع و دوباره وصل می کند.در تمامی سه نوع ساختار دستوری تعریف شده در بالا پارامتر ACL_NAME نام Acl های تعریف شده ای را مشخص می کند که معمولا نشانی IPکاربران در آن تعریف شده و توسط deny_info مورد پردازش قرار می گیرد.

زمانی که نتیجه ی رول http_access نمایش یک صفحه ی خطا باشد، اسکوئید آخرین Aclمورد ارزیابی قرار گرفته در

رولهای http_access را به خاطر می سپارد و درصورتکیه در آنها برچسب deny_info وجود داشته باشد اسکوئید صفحه خطای ایجاد شده را تحویل می دهد.به مثال زیر توجه کنید:

خطوط زیر را به عنوان تنظیمات در نظر بگیرید:

```
acl example_com dstdomain .example.com
acl example_net dstdomain .example.net
acl png_file    urlpath_regex -i \.png$
http_access allow example_net
http_access deny example_com png_file
http_access deny all
deny_info TCP_RESET example_com
deny_info http://example.net/ png_file
```

در اینجا فرض کنیم کاربری می خواهد از وب سایت http://www.example.com/default.png. بازدید کند مطابق تنظیمات بالا ، رول اول دسترسی با نشانی example_net مطابقت ندارد بنابراین اسکوئید نشانی مورد نظر را با رول دوم مطابقت می دهد در اینجا نشانی مورد نظر با Aclهای example_comوpng_file مطابقت دارد بنابراین توجه کنید که اسکوئید در اینجا Aclقبلی را مورد ارزیابی قرار داده و نتیجه آن عدم مجوز دسترسی به png_file است بنابراین اسکوئید یک deny_info را که بتواند آن را با Acl نوع png_file تطبیق دهد جستجو می کند که نتیجه ی آن ارسال کد ۳۰۲ از طرف کاربر HTTP و هدایت درخواست کاربر به سوی نشانی http://example.net است.

حال با تغییر وضعیت Aclهای موجود در تنظیمات پیکربندی بالا ، خطوط پیکربندی مذکور به صورت زیر نوشته می شوند:

```
acl example_com dstdomain .example.com
acl example_net dstdomain .example.net
acl png_file    urlpath_regex -i \.png$
http_access allow example_net
#Notice the switch
http_access deny png_file example_com
```

http_access deny all

deny_info TCP_RESET example_com

deny_info http://example.net/ png_file

حال در صورتکیه کاربری نشانی http://www.example.com/default.png را درخواست کند نتیجه ی آن قطع ارتباط TCPو وصل مجدد آن (راه اندازی دوباره ی ارتباط) است که دلیل آن عملکرد Acl ی به نام example_com است و نه png_file.

همانطور که در این مثال مشاهده کردید دستور deny_infoبه طور عمده برای هدایت کاربران به سوی یک صفحه ی خطا طراحی شده است.

آشنایی با ابزارهای تغییر مسیر نشانی محبوب

تاکنون که چگونگی به کارگیری و عملکرد ابزارهای تغییر مسیر نشانی های URL و همچنین ارتباط آنها را اسکوئید را آموختیم حال می خواهیم با برخی از محبوب ترین و بهترین ابزارهای تغییر مسیر نشانی های URLآشنا شویم.جهت مشاهده فهرست کلیه ی ابزارهای موجود می توانید به صفحه ی http://www.squid-cache.org/Misc/related-software.html مراجعه نمائید.

نرم افزارSquidGuard

SquidGuard نرم افزاری است که ترکیبی از ابزارهای فیلترینگ ، بازنویسی نشانی های URL،کنترل سطح دسترسی را به همراه دارد.از جمله خصوصیات بارز این نرم افزار میتوان به سرعت بالای پردازش ، رایگان بودن ،قابلیت انعطاف بالا و راه اندازی آسان اشاره کرد.موارد زیر بخشی از توانایی های این نرم افزار قدرتمند را بیان می کند:

- ایجاد محدودیت برای دسترسی گروهی از کاربران به نشانی های URLبه خصوص و یا وب سرورهای آنها
- محدودیت دسترسی برای گروهی ار کاربران تعریف شده در لیست سیاه
- هدایت URLهای مسدود شده به سوی صفحات حاوی اطلاعات مفید در زمینه دلایل مسدود بودن آنها
- هدایت کاربران ناشناس به سوی صفحات از قبل تعریف شده
- و موراد بسیار دیگر...

جهت آشنایی بیشتر با این نرم افزار می توانید به وب سایت رسمی آن به نشانی http://www.squidguard.org مراجعه نمائید.

نرم افزار Squirm

Squirm یکی دیگر از ابزارهای سریع و قابل پیکربندی است که برای مواردی چون بازنویسی نشانی های URL همراه با اسکوئید به کار می رود.جهت کسب اطلاعات دقیق تر د رمورد آن می توانید به نشانی http://squirm. foote.com.au مراجعه نمائید.برخی از ویژگیهای Squirm شامل موارد زیر است:

- بسیار سریع است و مصرف حافظه ی بسیار پایینی دارد
- قادر به خواندن و اعمال تغییر تنظیمات پیکربندی هنگام اجرا و بدون نیازبه راه اندازی مجدد است
- قادر به اجرا در حالت bypass است زمانی که فایل پیکربندی با اشکال مواجه شود
- موجود فضای تعاملی در آن جهت آزمایش تنظیمات پیکربندی جدید

نرم افزار Ad Zapper

Ad Zapper نیز یکی دیگر از ابزارهای محبوب بازنویسی نشانی های URLاست به به منظور حذف تبلیغات و صفحات pop-up ، انیمیشن های موجود در وب سایتها و سایر موارد حذف محتوا به کار می رود.Ad Zapper از ساختار دستوری خاصی پیروی می کند که از آنها به منظور انجام موارد بالا استفاده می کند.جهت آشنایی با ساختار دستوری این ابزار به نشانی http://adzapper.sourceforge.net مراجعه فرمائید.

خلاصه فصل

در این فصل با انواع نرم افزارهای تغییر مسیر و بازنویسی نشانی های URLآشنا شدیم و نحوه عملکرد و ارتباط آنها با اسکوئید را مورد بررسی قرار دادیم این ابزارها به عنوان یک پردازش خارجی که با اسکوئید در تعامل هستند می توانند کارهایی را که اسکوئید قادر به انجام آنها نیست و یا پردازش سنگینی را به آن وارد می کنند را به راحتی انجام دهند.همچنین با دستور deny_info که به منظور هدایت کاربران به سوی صفحات خطا به کار می رودآشنا شدیم .به کارگیری این دستور موجب ایجاد یک فضای تعاملی میان کاربران و اسکوئید می شود به طوریکه با ایجاد یک صفحه ی خطای مناسب می توان کاربران را از علت نمایش آن آگاه کرد و حتی راهنمایی های لازم را جهت برطرف کردن آن برای کاربر نمایش داده شود.

به طور کلی مطالب زیر در این فصل شرح داده شدند:

- آشنایی با ابزارهای تغییر مسیر نشانی های URLو چگونگی به کارگیری آنها
- چگونگی ارتباط و تعامل اسکوئید با این ابزارها
- ایجاد ابزار تغییر مسیر و بازنویسی نشانی به صورت اختصاصی
- پیکربندی اسکویئد به منظور ارتباط و استفاده از ابزارهای تغییر مسیر نشانی های URL
- آشنایی مختصر با محبوبترین ابزارهای تغییر مسیر نشانی که موجب صرفه جویی در مصرف پهنای باند و کنترل بیشتر بر محتوای شبکه می شوند.

تمامی مطالب فصلهای گذشته به چگونگی به کارگیری اسکوئید و دستورات مختلف پیکربندی آن اختصاص یافت.حال زمان آن رسیده است که با چگونگی رفع اشکال اسکوئید و آمادگی برای مقابله با آنها بیشتر آشنا شویم.در فصل آینده به طور اختصاصی به چگونگی رفع اشکال و خطایابی اسکوئید خواهیم پرداخت.

فصل دوازدهم

راهنمای رفع اشکال اسکوئید

در طول مطالب مطرح شده در فصلهای مختلف کتاب ، با چگونگی راه اندازی و پیکربندی اسکوئید از طریق روش های گوناگون ، چگونگی بهینه سازی آن و دستورات ایجاد شده به منظور به کارگیری حداکثر توانایی های اسکوئید آشنا شدیم.تا کنون توانستیم از تمامی قابلیتهای اسکوئید استفاده کرده و آن را در شبکه ی خود پیاده سازی کنیم. در برخی موارد و به دلیل مشکلات ناشی از عدم پیکربندی صحیح و یا دیگر اشکالات پیش آمده مانند وجود اشکال در اجزاء سخت افزاری پراکسی سرور ، اسکوئید عملکرد درستی ندارد و از ادامه فعالیت باز می ایستد و یا عملکرد آن به مشکلات جدی همراه می شود.در این فصل می خواهیم با رایجترین مشکلاتی که اسکوئید با آن مواجه می شود آشنا شده تا در مواردی که با این مسائل مواجه شدیم ، عملکرد صحیحی داشته باشیم.

در این فصل مطالب زیر عنوان خواهد شد:

- آشنایی با مشکلات رایج
- رفع اشکال مشکلات ایجاد شده
- استفاده از مستندات آنلاین و گزارش اشکالات به وجود آمده

برخی از مشکلات رایج

بسیاری از مشکلات به وجود آمده ناشی از عدم پیکربندی صحیح ویا وجود دستورات مبهم در تنظیمات اسکوئید است همچنین برخی از خطاهای ناشناخته نیز ممکن است ناشی از وجود یک باگ ناشناخته در سیستم عامل یا اسکوئید باشد.برای اینکه بتوانید به راحتی این مشکلات را شناسایی و حل کنید باید قبل از مواجه با هرگونه مسئله ای با این مشکلات آشنا بوده و راههای رفع آنها را فراگرفته باشید .در ادامه با برخی از خطاهای رایج و راههای رفع آن ها آشنا می شویم.

عدم توانایی اسکوئید در نوشتن و ثبت فایلهای ثبت رخداد

گاهی اوقات ممکن است با راه اندازی مجدد اسکوئید با پیغام هشداری مانند زیر در فایل cache.log مواجه شویم:

```
WARNING: Cannot write log file: /opt/squid/var/logs/cache.log
/opt/squid/var/logs/cache.log: Permission denied
  messages will be sent to 'stderr'.
```

این مسئله معمولا زمانی پیش می آید که نام کاربری ای که در سیستم عامل اقدام به راه اندازی اسکوئید کرده است از مجوز لازم و کافی جهت دسترسی و نوشتن در فایلهای ثبت رویداد برخوردار نیست.این مشکل معمولا هنگام نصب اسکوئید در حالت کامپایل به وجود می آید و در صورت استفاده از بسته های باینری و آماده ی موجود در توزیعهای مختلف لینوکس ،ابزار نصاب بسته ها خود مجوزهای لازم جهت خواندن و نوشتن در این فایلها را تنظیم کرده و از بروز مشکلات این چنینی جلوگیری به عمل می آورد.این مشکل به راحتی و با تغییر مالکیت دایرکتوری فایلهای ثبت رخداد حل می شود.اسکوئید هنگام راه اندازی از طریق نام کاربری nobody که در دستور cache_effective_user تعیین می شود راه اندازی می شود که این نام کاربری در اکثر موارد در سیستم عامل تعریف نشده و این موضوع موجب توقف فعالیت نوشتن در فایلها می شود.با استفاده از دستور chown به صورت زیر این مشکل برطرف می شود:

```
chown –R nobody:nobody /opt/squid/var/logs/
```

پس از اجرای دستور بالا در ترمینال با راه اندازی مجدد اسکوئید مشکل ایجاد شده برطرف خواهد شد.

عدم توانایی در تعیین نام میزبان

خطای رایج دیگری که معمولا با آن مواجه می شویم به صورت زیر است:

```
FATAL: Could not determine fully qualified hostname.  Please set
'visible_hostname'
Squid Cache (Version 3.1.10): Terminated abnormally.
```

این مشکل زمانی به وجود می آید که هیچ مقدار مشخصی برای دستور visible_hostname در فایل پیکربندی تعیین نشده باشد .توجه داشته باشید که در نسخه ۳.۲ اسکوئید به بعد سطح اهمیت این خطا از نوع مهلک (Fatal) به نوع هشدار (Warning) تغییر یافته و برای جلوگیری از توقف فعالیت اسکوئید،نام میزبان به صورت خودکار localhost، تعیین می شود.این مشکل به راحتی و با تعیین مقدار برای دستور visible_hostname در فایل پیکربندی به صورت زیر برطرف می شود:

```
visible_hostname proxy.example.com
```

در صورتکیه مانند مثال بالا آدرس دامنه به کار رفته باشد ،باید از یک DNS سرور به منظور تبدیل نام دامنه به آدرس ip استفاده شود همچنین در گروهی از پراکسی سرور ها که به صورت کلاستر با یکدیگر در ارتباط هستند برای هر سرور باید از یک نام منحصر به فرد استفاده شود.

عدم توانایی در ساخت دایرکتوریهای مخزن کش سرور

هنگامی که می خواهیم دایرکتوریهای جدیدی را به مخزن کش سرور خود اضافه کنیم ممکن است با خطای زیر مواجه شویم:

```
[root@saini ~]# /opt/squid/sbin/squid -z
2010/11/10 00:42:34| Creating Swap Directories
FATAL: Failed to make swap directory /opt/squid/var/cache: (13)
Permission denied
[root@saini ~]#
```

همانطور که از متن خطای به وجود آمده پیداست ،اسکوئید مجوز لازم برای ساخت دایرکتوریهای جدید را در اختیار ندارد.می توان این مشکل را به راحتی و از طریق تغییر ساخت دایرکتوری اصلی به صورت دستی و تغییر مالکیت آن به صورت زیر حل کرد:

```
mkdir /opt/squid/var/cache
chown nobody:nobody /opt/squid/var/cache
```

پس از وارد کردن دستورات بالا ،دستور قبل را مجددا اجرا کرده و در صورت موفقیت آمیز بودن فرایند ساخت دایرکتوریها پیغام زیر در خروجی نمایان می شود:

```
[root@saini etc]# /opt/squid/sbin/squid -z
2010/11/10 00:44:16| Creating Swap Directories
2010/11/10 00:44:16| /opt/squid/var/cache exists
2010/11/10 00:44:16| Making directories in /opt/squid/var/cache/00
2010/11/10 00:44:16| Making directories in /opt/squid/var/cache/01
...
```

عدم توانایی در بررسی دایرکتوریهای کش

گاهی اوقات زمانی که قابلیت کش کردن داده ها در اسکوئید فعال است ممکن است با خطایی مانند زیر مواجه شوید:

```
2010/11/10 00:33:56| /opt/squid/var/cache: (2) No such file or directory
FATAL: Failed to verify one of the swap directories, Check cache.log
for details. Run 'squid -z' to create swap directories
if needed, or if running Squid for the first time.
Squid Cache (Version 3.1.10): Terminated abnormally.
```

دلایل ایجاد این خطا شامل موارد زیر است:

- را ه اندازی اسکوئید برای اولین بار بدون ساخت دایرکتوریهای کش
- راه اندازی اسکوئید پس از بررسی دستور cache_dir

در فایل پیکربندی و وجود اشکال در مسیرهای دایرکتوری کش خطای بالا با اجرای دستور squid -z

برطرف خواهد شد.به طور کلی برای جلوگیری از بروز چنین مشکلاتی پس از هر بار اضافه کردن دایرکتوریهای جدید دستور

squid -z را اجرا کنید تا خطاهای احتمالی پیش آمده از همان ابتدا مشخص شوند.

آدرس مورد نظر در حال استفاده است

خطای رایج دیگری که معمولا با آن مواجه می شویم با عباراتی همچون Address already in use, Cannot bind socket, or Cannot open HTTP port به صورت زیر ظاهر می شود:

2010/11/10 01:04:20 |commBind: Cannot bind socket FD 16 to [::]:8080:

)98 (Address already in use

FATAL: Cannot open HTTP Port

Squid Cache (Version 3.1.10): Terminated abnormally.

این خطا زمانی پیش می آید که اسکوئید در زمان اجرا درتلاش برای ارتباط با درگاه تعیین شده در دستور http_port بر می آید حال اگر برنامه ی دیگری در حال استفاده از آن درگاه باشد اسکوئید قادر به برقراری ارتباط با آن نبوده و خطای بالا به وجود می آید.اولین مرحله برای حل این مشکل پیدا کردن برنامه ای است که در حال گوش کردن به آن درگاه است.این پروسه یعنی یافتن برنامه ی مورد نظر در سیستم عاملهای مختلف متفاوت است.روش های زیر در مورد سیستم عاملهای پرطرفدار کاربرد دارد.

سیستم عاملهای لینوکسی

برای یافتن برنامهمورد نظر در سیستم عاملهای لینوکسی دستور زیر را اجرا کنید:

lsof –i :8080

البته فراموش نکنید که درگاه 8080 را با درگاه مورد نظر خود عوض کنید.

برای توزیع های OpenBSD و NetBSD

در این توزیع ها از دستور fstat

به صورت زیر استفاده می کنیم:

fstat | grep 8080

خروجی دستور بالا فهرستی از تمامی اتصالات برقرار شده به سوی درگاه ۸۰۸۰ را نشان می دهد.

برای توزیعهای FreeBSD and DragonFlyBSD

برنامه ی مورد استفاده در این توزیعها جهت یافتن اتصالات برقرار شده به سوی یک درگاه مشخص، sockstat

نام دارد که طریقه ی به کارگیری آن مانند مثال زیر است:

```
sockstat -4l | grep 8080
```

این دستور تمامی برنامه های درحال اجرا که از درگاه ۸۰۸۰ استفاده می کنند را نشان می دهد.

روش های ذکر شده در بالا راههای پیدا کردن برنامه ی ای که با درگاهی که اسکوئید از آن استفاده می کند تداخل دارد به کار می رود جال پس از پیدا کردن برنامه ی مورد نظر راه حال های زیر برای حل مشکل توصیه می شود:

- درصورتکیه برنامه ی مورد نظر مهم بوده و به آن نیاز داریم بهترین راه تغییر درگاه http_port در فایل پیکربندی اسکوئید و راه اندازی مجدد آن است .
- بستن برنامه ی در حال استفاده از درگاه مورد نظر و راه اندازی مجدد اسکوئید است این راه حل موقتی است و با اجرای برنامه دوباره تداخل پیش می آید.

عملکرد اسکوئید کند شده است

دلیل رخ دادن این مسئله وارد آمدن فشار بیش از حد به سروری است که اسکوئید برروی آن راه اندازی شده است مثلا در صورتکیه مقدار تعیین شده در دستور cache_mem که میزان حافظه ی موقت اختصاص داده شده به اسکوئید را تعیین می کند بیش از حد بزرگ باشد و حافظه ی کافی برای اجرای سایر پروسه ها و برنامه های موجود برروی سیستم عامل باقی نمانده باشد ،سیستم عامل با کمبود حافظه مواجه شده و باعث تاخیر در اجرای برنامه های مختلف و مرتبط با اسکوئید و در نتیجه کاهش بازده اسکوئید می شود.

برای حل این مشکل انجام مراحل زیر پیشنهاد می شود:

- میزان حافظه ی مورد نیاز به منظور عملکرد صحیح و بدون نقص سیستم عامل بررسی شود و میزان حافظه ی تعیین شده برا ی دستور cache_mem متناسب با آن تعیین شود البته همواره میزان حافظه ی تعیین شده برای سیستم عامل را حدود ۵ تا ۱۰ درصد بیش از مقدار مورد نیاز برای اضطراری در نظر بگیرید.

- غیر فعال کردن Memory Pool با استفاده از دستور memory_pools off که این کار باعث آزاد شدن مقدار زیادی از حافظه ی مصرف شده توسط اسکوئید می شود.

- اسکوئید فهرستی از تمامی داده های کش شده را در حافظه ی رم نگهداری می کند در صورتکیه حجم دایرکتوریها و داده های کش شده زیاد باشد ،فهرست مورد نظر میزان قابل توجهی از حافظه ی رم را اشغال می کند در صورتیکه دو روش قبلی مشکل کمبود حافظه را حل نکنند ،باید میزان دایرکتوریهای کش سرور را کاهش دهیم تا حافظه ی کمتری را اشغال کنند.

درخواست و یا پاسخ بسیار بزرگ است

گاهی اوقات کاربران شبکه به طور مرتب پیغام خطای "The request or reply is too large" را دریافت می کنند.دلیل بروز این خطا این است که میزان حجم هدر های درخواست ،یا هدر های پاسخ درخواست از مقدار مجاز تعیین شده در تنظیمات پیکربندی اسکوئید فراتر رفته است .علت این مشکل با مقادیر موجود در دستورات request_header_max_size, request_body_max_size, reply_header_max_size, and reply_body_max_size مرتبط است و با تغییر مقادیر موجود در این دستورها و راه اندازی مجدد مشکل یاد شده برطرف می شود.

پیغام عدم دسترسی برروی سرور پراکسی

گاهی اوقات مشکلات پیش آمده بسیار گیج کننده و مبهم هستند به طوریکه همه چیز درست به نظر می رسد ولی مشکل همچنان با برجاست یکی از این مشکلات به ظاهر ساده ،عدم دسترسی به شبکه برروی سرور اصلی ای که اسکوئید برروی آن در حال اجراست به طورکیه تمام کاربران از طریق پراکسی سرور شما به اینترنت دسترسی دارند ولی شما برروی سرور اصلی این اجازه را ندارید! این مشکل زمانی پیش می آید که در فایل پیکربندی به آدرس IP سروری که اسکوئید برروی آن در حال فعالیت است اجازه ی دسترسی به شبکه را نداده اید.با بررسی ACL نوع localhost در تنظیمات پیکربندی و اضافه کردن IP سرور مورد نظر و نوشتن Access Control مناسب ،و در آخر راه اندازی مجدد اسکوئید مشکل ساده شده به راحتی رفع می شود.

رفع اشکال خطاهای ناشناخته و پیچیده

در بسیاری از موارد علت وقوع مشکلات و خطاهای پیش آمده ،وجود مشکلاتی در تنظیمات پیکربندی و یا محدودیت های سیستم عامل می باشد که با برسی دقیق تنظیمات اشکال پیش آمده برطرف می شود.اما در برخی مواردعلت اشکال پیش آمده نامشخص بوده و ما تاکنون تجربه ی مواجه با چنین مشکلی را نداشته ایم و امکان شناسایی آن از طریق بررسی فایلهای ثبت

رخداد cache.log وجود ندارد.سرویس دهنده ی اسکوئید به طور پیش فرض فقط اطلاعات ضروری در مورد وضعیت خود را در فایل cache.log ثبت می کند که این اطلاعات در مواردی که سطح پیچیدگی مشکل بالاست اصلا کافی نیست و ما برای حل مشکلات این چنینی به اطلاعات جامع تری در مورد فعالیتهای اسکوئید نیازمندیم.بدین منظور با استفاده از دستور debug_options در فایل پیکربندی می توانیم سطح اطلاعات و جزئیات ارائه شده در فایلهای سطح رخداد را گسترش دهیم.فرمت کلی دستور debug_options به صورت زیر است:

debug_options rotate=N section,verbosity [section,verbosity...]

پارامتر(rotate=N) rotateتعداد فایلهای ثبت رخداد cache.log را که هنگام اجرای دستور logrotate که به منظور گرد کردن و تکه تکه کردن فایلهای ثبت رخداد و به منظور کاهش حجم آنها و جلوگیری از هدر رفتن دیسک سخت در حالت یکپارچه نگهداری می کند را تعیین می کند که مقدار پیش فرض آن ۱ است.پارامتر section که یک عدد صحیح است در واقع یک جزء خاص از ترکیبات اطلاعات ارائه شده توسط اسکوئید را مشخص می کند و در واقع هر عدد نماینده ی یک معنای خاص برای اسکوئید است که این مقدار می تواند ALL به معنی تمامی جزئیات ممکن باشد.پارامتر verbosity هم مقدار عددی با معنی برای اسکوئید است که به نوعی اطلاعاتی برای هر یک از اجزاء section را با توجه به سطح انتخاب شده بیان می کند.در جدول زیر با این اعداد و معانی آنها آشنا می شویم:

توضیحات	سطح اطلاعات ارائه شده
فقط پیام های بحرانی ثبت می شوند.	0
هشدار ها و مشکلات مهم ثبت می شوند.	1
در سطح توضیحاتی و اطلاعاتی 2 ،پیام های هشدار &مشکلات سطحی و مشکلات بحرانی ثبت می شوند	2
تقریبا هرتوضیحات و اطلاعاتی که برای حل مشکلات مفید واقع می شوند در سطح 5 ثبت می شود.	3-5
در سطوح توضیحاتی و اطلاعاتی بالاتر از 5 کوچکترین فعالیت های ایجاد شده به همراه کاملترین سطح توضیحاتی ای که ممکن است ارائه شود در فایل ثبت رخداد ذخیره می شود.	6-9

تنظیمات پیکربندی پیش فذض این بخش به صورت زیر است:

debug_options rotate=1 ALL,1

در تنظیمات پیش فرض سطح توضیحات ارئه شده برای تمامی بخش های اطلاعاتی اسکوئید یک تعیین شده است که به معنای کمترین سطح اطلاعات و جزئیات ارائه شده در فایل ثبت رخداد است.

شماره گروه ها (Section Number) معمولا از طریق مشاهده کد منبع فایلهای ثبت رخداد مشخص می شود در بسیاری از موارد با مشاهده توضیحات ارارئه شده در کد فایلها می توان به این مقدار پی برد برای مثال توضیحات ارائه شده در فایل access_log.cc به صورت زیر است:

```
/*
...
 * DEBUG: section 46    Access Log
...
*/
```

این توضیحات به ما می گوید که شماره گروه تعیین شده برای Access log 46 است برای مشاهده فهرستی از شماره های گروه ارائه شده توسط اسکوئید به مسیر doc/debug-sections.txt واقع در کد منبع اسکوئید مراجعه کنید.جدول زیر فهرستی از مهمترین کد گروههای ارائه شده همراه با اسکوئید نسخه ی 3.1.10 را نشان می دهد:

شماره گروه	اجزا مرتبط با شماره گروه در اسکوئید
0	Announcement Server, Client Database, Debug Routines, DNS Resolver Daemon , UFS Store Dump Tool
1	Main Loop, Startup
2	Unlink Daemon
3	Configuration File Parsing, Configuration Settings
4	Error Generation

6	Disk I/O Routines
9	File Transfer Protocol (FTP(
11	Hypertext Transfer Protocol (HTTP(
12	Internet Cache Protocol (ICP(
14	IP Cache, IP Storage, and Handling
15	Neighbor Routines
16	Cache Manager Objects
17	Request Forwarding
18	Cache Manager Statistics
20	Storage Manager, Storage Manager Heap-based replacement, Storage Manager Logging Functions, Storage Manager MD5 Cache Keys, Storage Manager Swapfile Metadata, Storage Manager Swapfile Unpacker, Storage Manager Swapin Functions, Storage Manager Swapout Functions, Store Rebuild Routines, Swap Dir base object
23	URL Parsing, URL Scheme parsing
28	Access Control
29	Authenticator, Negotiate Authenticator, NTLM Authenticator
31	Hypertext Caching Protocol
32	Asynchronous Disk I/O
34	Dnsserver interface
35	FQDN Cache
44	Peer Selection Algorithm

46	Access Log
50	Log file handling
51	Filedescriptor Functions
55	HTTP Header
56	HTTP Message Body
57	HTTP Status-line
58	HTTP Reply (Response)
61	Redirector
64	HTTP Range Header
65	HTTP Cache Control Header
66	HTTP Header Tools
67	String
68	HTTP Content-Range Header
70	Cache Digest
71	Store Digest Manager
72	Peer Digest Routines
73	HTTP Request
74	HTTP Message
76	Internal Squid Object handling
78	DNS lookups, DNS lookups; interacts with lib/rfc1035.c
79	Disk IO Routines, Squid-side DISKD I/O functions, Squid-side Disk I/O functions , Storage Manager COSS Interface, Storage Manager UFS Interface
84	Helper process maintenance

89	NAT / IP Interception
90	HTTP Cache Control Header, Storage Manager Client-Side Interface
92	Storage File System

با توجه به اطلاعات مربوط به جدول می خواهیم به طور مثال مشکلات احتمالی پیش آمده در پروتکل HTTP را با ارائه کاملترین جزئیات و اطلاعات ممکن بررسی کنیم با توجه به شماره گروه 11 در جدول بالا تنظیمات پیکربندی به صورت زیر ارائه می شود:

debug_options ALL,1 11,5

پس از اضافه کردن خط بالا به تنظیمات پیکربندی به منظور اعمال تغییرات اسکوئید باید یک بار راه اندازی مجدد شود سپس در صورتکیه در مرورگر خود آدرس www.example.com را وارد کنیم خطوط گزارش مشابه خطوط زیر فایل ثبت رخداد cache.log ظاهر می شود البته در خطوط زیر به منظور کوتاه تر شدن خطوط گزارش گزارشات مربوط به زمان ایجاد رخداد ها را حذف کرده ایم:

```
httpStart: "GET http://www.example.com/"
http.cc(86) HttpStateData: HttpStateData 0x8fc4318 created
httpSendRequest: FD 14, request 0x8f73678, this 0x8fc4318.
The AsyncCall HttpStateData::httpTimeout constructed, this=0x8daead0
[call315]
The AsyncCall HttpStateData::readReply constructed, this=0x8daf0c8
[call316]
The AsyncCall HttpStateData::SendComplete constructed, this=0x8daf120
[call317]
httpBuildRequestHeader: Host: example.com
httpBuildRequestHeader: User-Agent: Mozilla/5.0 (X11; U; Linux i686;
en-US; rv:1.9.2.3) Gecko/20100403 Fedora/3.6.3-4.fc13 Firefox/3.6.3
```

```
GTB7.1
httpBuildRequestHeader: Accept: text/html,application/
xhtml+xml,application/xml;q=0.9,*/*;q=0.8
httpBuildRequestHeader: Accept-Language: en-us,en;q=0.5
httpBuildRequestHeader: Accept-Encoding: gzip,deflate
httpBuildRequestHeader: Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
httpBuildRequestHeader: Keep-Alive: 115
httpBuildRequestHeader: Proxy-Connection: keep-alive
httpBuildRequestHeader: If-Modified-Since: Fri, 30 Jul 2010 15:30:18
GMT
httpBuildRequestHeader: If-None-Match: "573c1-254-48c9c87349680"
comm.cc(166) will call HttpStateData::SendComplete(FD 14,
data=0x8fc4318) [call317]
entering HttpStateData::SendComplete(FD 14, data=0x8fc4318)
AsyncCall.cc(32) make: make call HttpStateData::SendComplete [call317]
...
```

همانطور که می بینید اسکوئید در این حالتکاملترین اطلاعات ممکن را در اختیار ما قرار می دهد که با دقت در خطوط به راحتی می توان فرایند ساخت و تبادل هدر های HTTP مورد نیاز و همچنین سایر اطلاعات ضروری را مشاهده نمود.

رفع اشکال Access Control ها

به طور معمول با توجه به وسعت شبکه ،تعداد ACL های زیادی در فایل پیکربندی ساخته می شوند که هر کدام به منظور اعمال خاصی ایجاد شده اند گاهی اوقات به دلیل گستردگی بیش از حد این تنظیمات مشکلات ناخواسته ای از قبیل در دسترس نبودن قسمتی از شبکه ،ناتوانی کاربران درمشاهده برخی وب سایتها و مشکلات از این قبیل به وجود می آید که رفع اشکال همه ی آنها فقط با بررسی دوباره ی تنظیمات ممکن است به نتیجه ی مناسبی ختم نشود به همین خاطر می توان از ابزارهای موجود در اسکوئید مانند debug_optionsاستفاده نمود.مثال زیر چگونگی اشکال یابی access control ها را به ما نشان می دهد:

فرض کنیم خطوط زیر در فایل پیکربندی ما موجود باشد:

```
acl example dstdomain .example.com
acl png urlpath_regex -i \.png$
http_access deny png example
http_access allow localhost
http_access allow localnet
http_access deny all
```

با جستجو در جدول شماره ی گروه ها شماره مربوط به access control، 28 تعیین شده است که برای استفاده همراه با دستور debug_options به صورت زیر عمل می کنیم:

```
debug_options ALL,1 28,3
```

دراین دستور سطح توضیحات ارائه شده برای access control ها 3 یعنی جزئیات بیشتر و برای سایر گروه ها مقدار پیش فرض 1 در نظر گرفته شده است که برای اعمال این تنظیمات راه اندازی مجدد اسکوئید الزامی است حال با استفاده از مرورگر خود آدرس www.example.com/default/.png را جستجو می کنیم حال با مشاهده گزارشات موجود در cache.log خطوط مشابه زیر را مشاهده می کنیم:

```
...
1. ACLChecklist::preCheck: 0x8fa3220 checking 'http_access deny png
example'
2. ACLList::matches: checking png
3. ACL::checklistMatches: checking 'png'
4. aclRegexData::match: checking '/default.png'
5. aclRegexData::match: looking for '\.png$'
6. aclRegexData::match: match '\.png$' found in '/default.png'
7. ACL::ChecklistMatches: result for 'png' is 1
8. ACLList::matches: checking example
```

9. ACL::checklistMatches: checking 'example'
10. aclMatchDomainList: checking 'www.example.com'
11. aclMatchDomainList: 'www.example.com' found
12. ACL::ChecklistMatches: result for 'example' is 1
13. aclmatchAclList: 0x8fa3220 returning true (AND list satisfied)

...

در خط اول اسکوئید درخواست مورد نظر را مطابق تنظیمات موجود در http_access deny png example دنبال می کند در خط دوم اسکوئید پردازش مورد نظر را برای ACL png دنبال می کند د رخط چهارم به بعد اسکوئید درخواست موجود را با نوع رفتار تعیین شده در ACL png مطابقت می دهد و در سایر خطوط نیز در صورتکیه تطبیق نجام شده صحیح باشد با توجه به نوع رفتاری که در این مثال در نظر گرفته شده است اسکوئید اجازه ی دریافت درخواست توسط کاربر را با توجه به خط http_access deny png example واقع در تنظیمات پیکربندی را نخواهد داد.همانطور که مشاهده می کنید با استفاده از این نوع روش خطایابی می توان یک درخواست را از نقطه ی آغاز درخواست تا مقصد نهایی آن یعنی دریافت پاسخ توسط کاربر ردیابی کرد و در این میان تمامی اعمال و رفتارهایی که ممکن است توسط دستورات موجود در تنظیمات بر درخواست مورد نظر اعمال شود را به دقت زیر نظر داشت و مشکلات احتمالی در این مسیر را که احیانا ناشی از خطای تنظیمات پیکربندی باشد را شناسایی و رفع نمود که در این مثال 13 مرحله از این فرایند نشان داده شده است که هر مرحله عملیات خاصی را شامل می شود.

استفاده از راهنمایی های آنلاین و گزارش باگ های احتمالی

در صورتکیه با مطالعه ی مطالب این فصل باز هم نتوان مشکل پیش آمده را برطرف کردد ،بااستفاده از وب گاه http://www.squid-cache. org/Support/mailing-lists.html می توان مسائل و مشکلات خود را با سایر کاربران اسکوئید در میان گذاشت همچنین می توان از مسائل و راهنمایی هایی که در این صفحات موجود است استفاده کرد. یکی دیگر از مراجع مناسب یادگیری اسکوئید دانشنامه آنلاین اسکوئید است که همواره از طریق آدرس http://wiki.squid-cache.org قابل دسترسی است دراین دانشنامه ،بسیاری از سوالات رایجی که برای کاریران اسکوئید پیش می آید را طبقه بندی کرده و به آنها پاسخ داده شده است همچنین در بخش های مختلف آن مثال های بسیازی زیاد و متنوعی از قسمت های مختلف فایل پیکربندی اسکوئید مطرح شده است که می توان از آنها در پیکربندی اسکوئید استفاده کرد.و در آخر نیز در صورتکیه در حین استفاده و بررسی اسکوئید به باگهای احتمالی برخورد کردید می توانید از طریق آدرس http://bugs.squid-cache.org آن را با توسعه دهندگان و گروه های توسعه ی اسکوئید در میان بگذارید که البته این

کار خود دارای مراحل و شرایط خاصی است که در هنگام ارائه گزارش باید به آن توجه کرد از جمله:

- نسخه و شماره ی دقیق اسکوئید مورد استفاده
- نام سیستم عامل میزبان به همراه شماره ی نسخه
- در چه شرایطی باگ مورد نظر آشکار شده است
- در صورتکیه ایده ای برای حل این مشکل دارید آن را به طور واضح شرح دهید

با انجام اینکار علاوه بر استفاده از اسکوئید می توان در نگهداری ،توسعه و رفع اشکال این سرویس دهنده محبوب وقدرتمند مشارکت داشت.

خلاصه فصل

در این فصل با رایجترین مشکلات احتمالی اسکوئید آشنا شدیم همچنین راههای مختلف رفع مشکلات ایجاد شده و چگونگی به کارگیری فایل ثبت رخداد cache.log برای این منظور آشنا شدیم به طور کلی مطالب ارائه شده در این فصل شامل موارد زیر است:

- برخی از رایج ترین مشکلات موجود به همراه راههای برطرف نمودن آنها
- رفع اشکال قسمت های مختلف تنظیمات پیکربندی اسکوئید با استفاده از cache.log
- استفاده از مراجع آنلاین رفع اشکال اسکوئید
- فرایند گزارش یک باگ کشف شده توسط شما

ضمیمه

در این بخش می خواهیم با فایل پیکربندی اسکوئید به صورت عملی آشنا شویم. تنظیمات و دستوراتی که در ادامه مشاهده می کنید در طول فصلهای مختلف کتاب مورد بحث قرار گرفته اند که در اینجا در کنار هم گردآمده اند و تنظیمات اصلی پیکربندی اسکوئید را تشکیل داده اند.لازم به ذکر است گزینه هایی که در این تنظیمات وجود دارند به صورت پیش فرض بوده و شما باید آن ها را مطابق نیازهایتان ویرایش نمائید.در ادامه فایل اصلی پیکربندی اسکوئید را مشاهده می کنید:

```
acl test dstdom_regex
http_access deny test
acl time1 time SMTWHFA 07:00-12:00
acl time2 time SMTWHFA 12:01-19:00
acl time3 time SMTWHFA 19:01-23:59
#######################
acl game dstdom_regex -i joke
http_access deny joke time1
http_port 3128 intercept
http_port 3128 tproxy
http_port 8080
icp_port 3130
snmp_port 3401
dns_nameservers 4.2.2.4 8.8.8.8 4.2.2.1 8.8.4.4

quick_abort_min 1024 KB
quick_abort_max 4096 KB
quick_abort_pct 5

hierarchy_stoplist cgi-bin ?
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY

cache_mem 3000 MB
cache_swap_low 95
cache_swap_high 98

maximum_object_size 20 MB
minimum_object_size 0 KB
#######################
#pool directories
cache_dir aufs /var/spool/squid/cache 10000 16 256
#######################
#logs
coredump_dir var/spool/squid
cache_access_log /var/log/squid/access.log
cache_access_log none
cache_log /var/log/squid/cache.log
cache_log none
useragent_log /var/log/squid/useragent.log
cache_store_log /var/log/squid/store.log
cache_store_log none
logfile_rotate 2
log_uses_indirect_client off
strip_query_terms off
#######################
# Filtering
acl x url_regex http://www.example.net http://example.net www.example.net
http_access deny x
```

```
deny_info http://www.example.org x

acl blacklist dstdomain .example.orf
no_cache deny blacklist
acl block url_regex "/usr/local/squid/etc/blacklist.acl"
http_access deny blacklist
######################
#Advertising
acl image url_regex "/etc/squid/google"
deny_info http://0.0.0.0/zap/pic.gif image
http_access deny image
######################
acl snmppublic snmp_community public
snmp_access allow snmppublic
snmp_access allow all
snmp_incoming_address 0.0.0.0
snmp_outgoing_address 0.0.0.0
######################
# Refresh Patterns
refresh_pattern ^ftp: 144000 20% 1008000
refresh_pattern -i \.(gif|png|jpg|jpeg|ico|bmp)$ 260000 90% 260009 override-expire
refresh_pattern -i \.(iso|avi|wav|mp3|mp4|mpeg|swf|flv|x-flv|mpg|wma|ogg|wmv|asx|asf)$
260000 90% 260009 override-expire
refresh_pattern -i \.(deb|rpm|exe|zip|tar|tgz|ram|rar|bin|ppt|doc|tiff|pdf|uxx)$ 260000
90% 260009 override-expire
refresh_pattern -i \.index.(html|htm)$ 1440 90% 40320
refresh_pattern -i \.(html|htm|css|js)$ 1440 90% 40320
refresh_pattern (/cgi-bin/|\?) 0 0% 0
refresh_pattern . 0 20% 4

request_header_max_size 64 KB
negative_ttl 3 minutes
positive_dns_ttl 15 hours
pipeline_prefetch on
ipcache_size 20480
ipcache_low  94
ipcache_high 96
cache_mgr admin@example.com
cachemgr_passwd 2012 all
visible_hostname squid
cache_effective_user nobody
######################
#Access Control Lists
acl manager proto cache_object
acl localhost src 127.0.0.1
acl local src 192.168.0.0/16
acl SSL_ports port 443 563
acl Safe_ports port 80 20 21 443 70 210 8080 280 488 591 777 901 1024-65535
acl purge method PURGE
acl CONNECT method CONNECT
######################
#HTTP Access Rules
http_access allow manager localhost
http_access allow manager local
http_access deny manager
http_access allow purge localhost
http_access allow purge local
http_access deny purge
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
http_access allow local
http_access deny all
```

منابع:

Squid.Proxy.Server.3.1.Beginners.Guide
www.squid-cache.org
Wikipedia