توضیح اجمالی فرآیند:

فراخوانی باید به صورت نام پورت ، سیاست کش ، و اندازه کش انجام شود

./proxy 1111 2 500

دستور بالا به معنی فعال شدن پروکسی بر روی پورت 111 با سیاست دوم و با تا تعداد 500عدد فایل در کش می‌باشد.

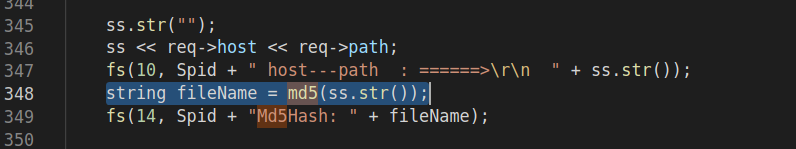
تابع fs نقش چاپ مقادیر با فرمت رنگی در صورت پشتیبانی ترمینال را دارد.

مدیریت فایل‌های کش شده توسط یه پردازه با نام فرضی cacheManager انجام می‌شود

در ابتدای برنامه یک پایپ pipe ایجاد شده سپس برنامه fork می‌شود و پردازه مدیرکش در حلقه خواندن از روی پایپ قرار می‌گیرد، به علت باز بودن نوشتن بر روی پایپ ، دستور خواندن بلاک کننده بوده و سپس بر روی پورت مربوطه Listen می کند و با رسیدن هر درخواست یک Fork برای رسیدگی به درخواست و پورت جدید ایجاد می‌کند و ..

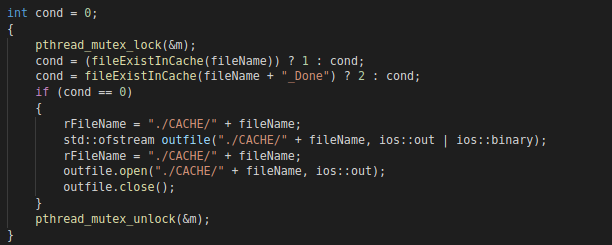
بعد از دریافت فایل ، یا ارسال آن به کلاینت به مدیرکش توسط پایپ اطلاع داده می‌شود تا در صورت لزوم بر اساس سیاست اتخاذ شده کش کردن ، فایل‌های اضافی را حذف کند.

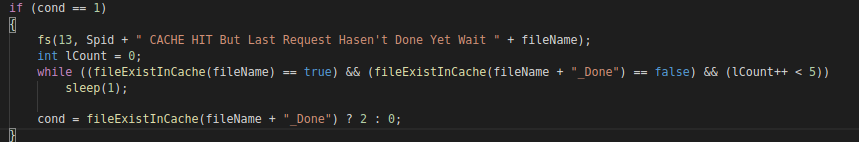
فایل‌ها در واقع پاسخ سرور به درخواست get پروکسی می‌باشند ، که عملا از هش کد MD5 آدرس URL مربوطه نام آن ها ساخته می‌شود.



فایل‌ها درون شاخه ./CACHE ذخیره می‌شوند

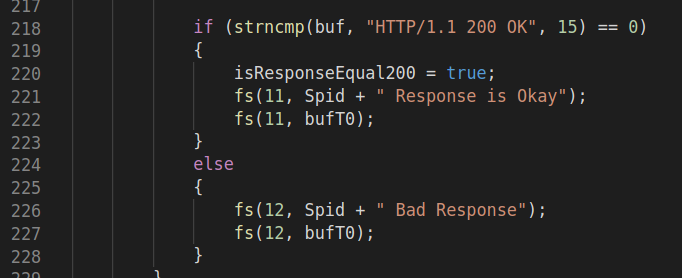
از یک قفل موتاکس استفاده شده تا در صورتی که دو درخواست یکسان همزمان به پروکسی رسید یکی از آن‌ها اقدام به دریافت فایل کند و درخواست دوم تا 5 ثانیه معطل می‌ماند.



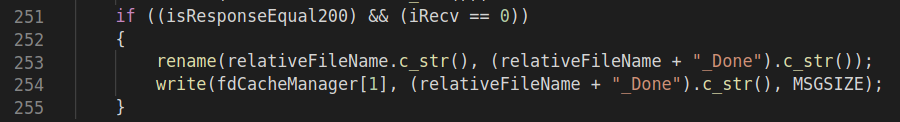


در حین کش کردن یک فایل اطلاعات به سمت کلاینت نیز ارسال می‌شود و با پایان آن به نام فایل مقدار \_\_Done اضافه می‌شود.

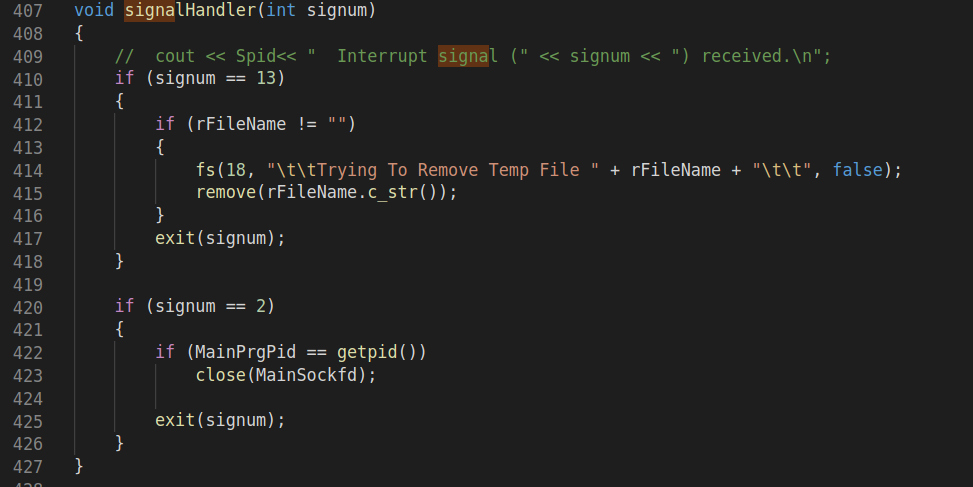
در صورتی که پاسخ سرور شامل HTTP/1.1 200 ok باشد فایل‌ها ذخیره خواهند شد . دو پیام Response is okay و Bad Response و مقداری از محتوای پاسخ نیز بر صفحه نمایش داده می‌شود.



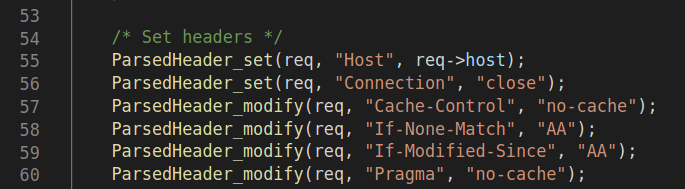
در صورت موفق بودن دریافت فایل به مدیرکش از طریق پایپ اطلاع داده می‌شود



همچنین سیگنال 13 مربوط به ارسال بر روی سوکت بسته شده نیز مدیریت شده تا در صورت لغو عمل دریافت فایل توسط کلاینت در صورت امکان فایل نیمه کاره‌ی کش شده حذف شود

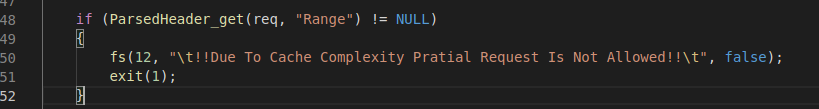


\*\*\* درخواست های رسیده از مرورگر (درخواست‌های get کلاینت) تغییر کرده و هدر آن تغییر می‌کند.

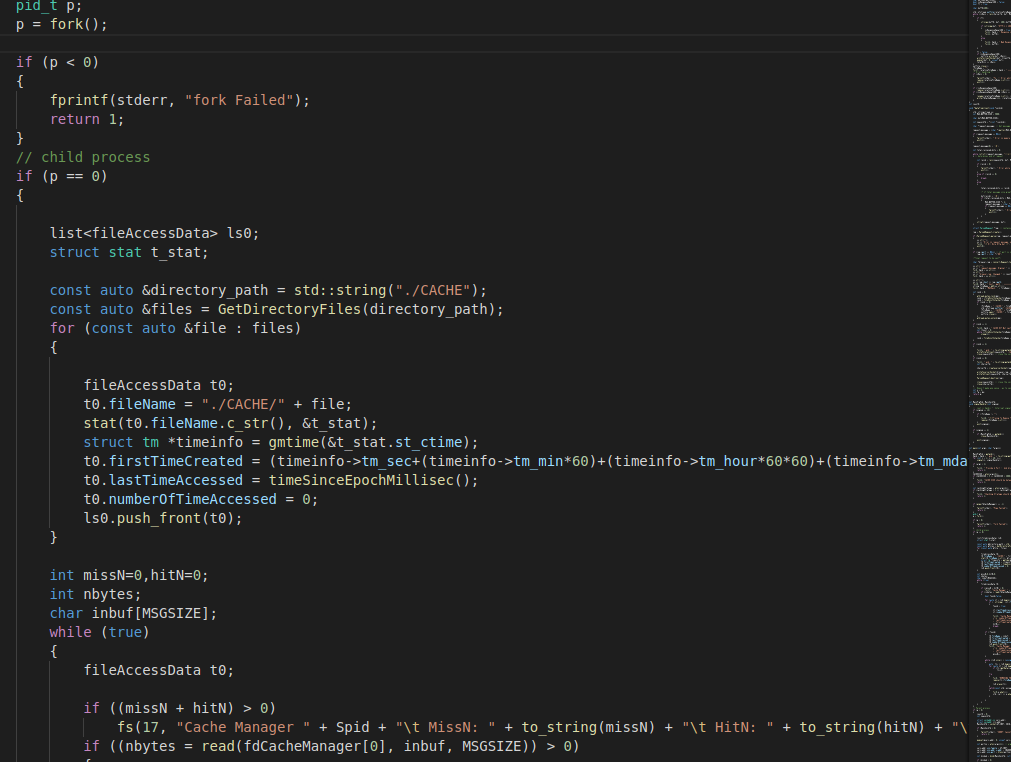


اطلاعات عدم کش به آن اضافه می‌شود تا سرور دوباره فایل را ارسال کند

همچنین در صورتی که درخواست get شامل Range باشد (مثلا در زمان دانلود یک فایل مرورگرها چندین کانکشن برای دریافت سریعتر فایل باز می‌کنند) آن درخواست لغو می‌شود

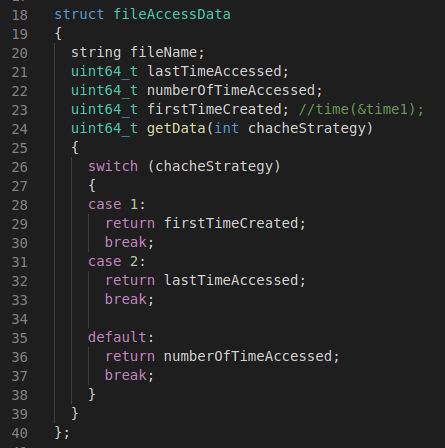


Fork اول جهت ایجاد مدیرکش با توجه به سیاست مربوطه



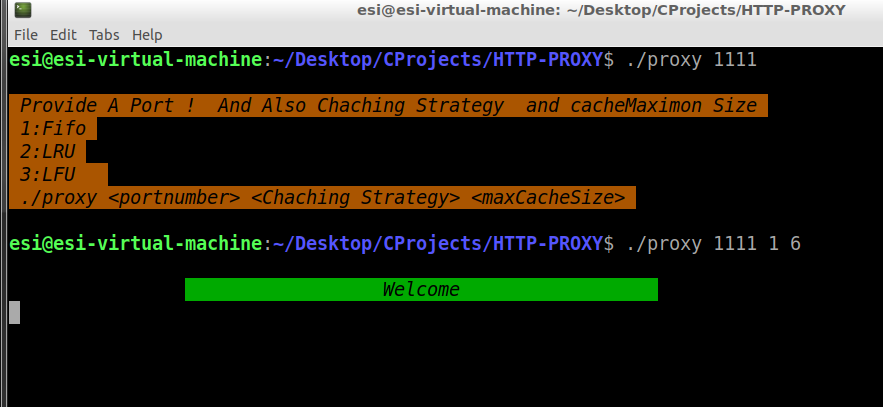
ابتدا همه‌ی اطلاعات فایل‌های داخل کش با ساختمان داده مربوطه اضافه شده و زمان ساخت آن‌ها (این زمان جهت مقایسه می‌باشد در ادامه هم فرمت نیست اما مشکلی ایجاد نمی‌کند) نیز اضافه می‌شود.

ساختمان داده مربوطه

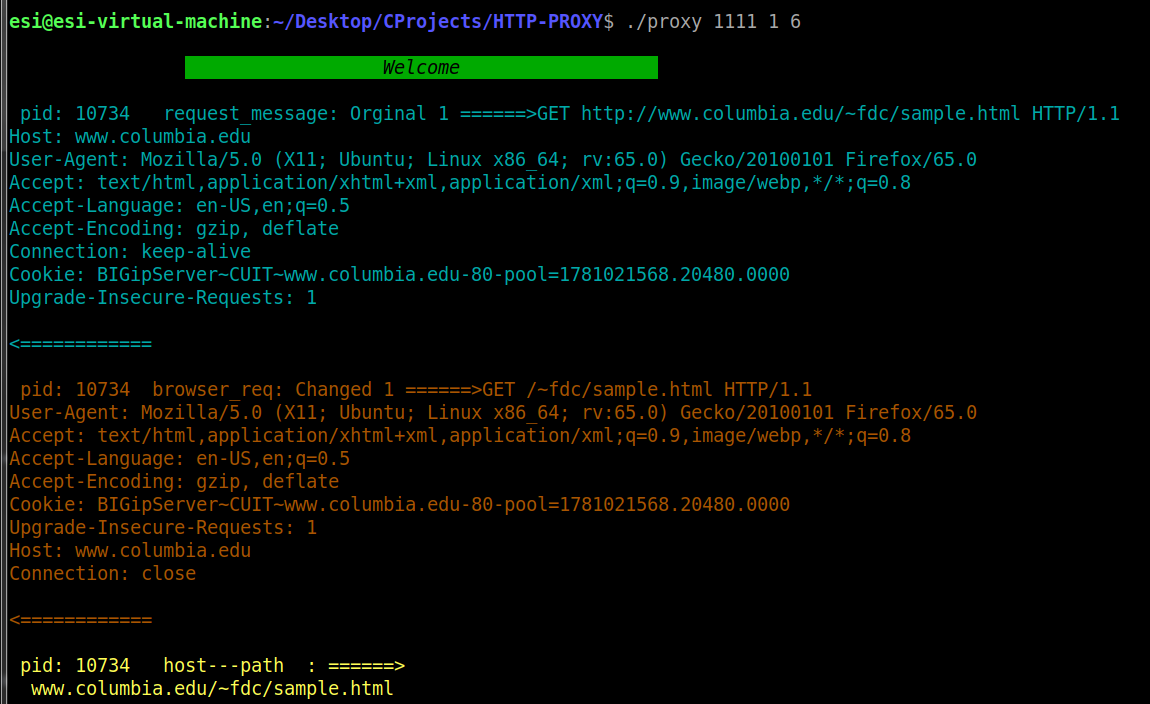


اطلاعات مروبط به درصد cach hit & miss نیز بر صفحه چاپ می‌شود .

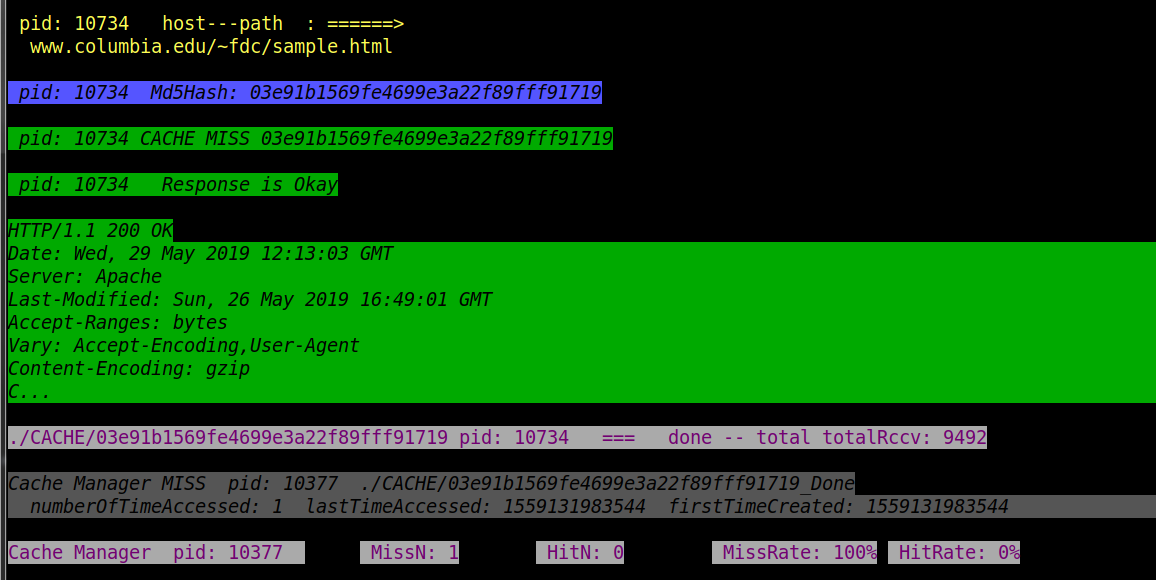
تصاویری از ترمینال و اجرا:



درخواست اولیه و تغییر کرده و آدرس درخواستی:

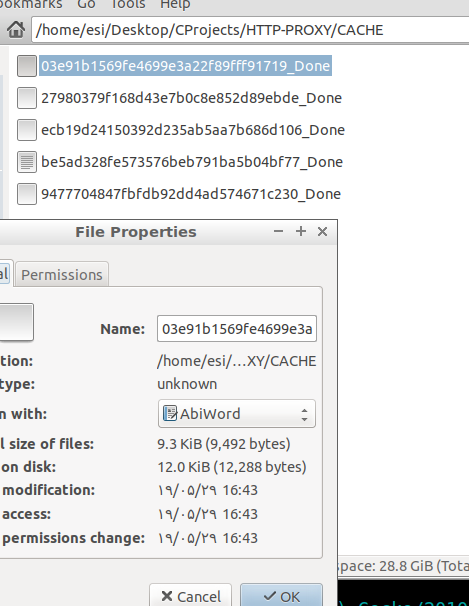


کد هش آدرس و دریافت اوکی بودن پاسخ و حجم فایل دریافتی و اطلاعات خاکستری رنگ توسط پایپ به مدیر کش رسیده و جزییات میس و هیت

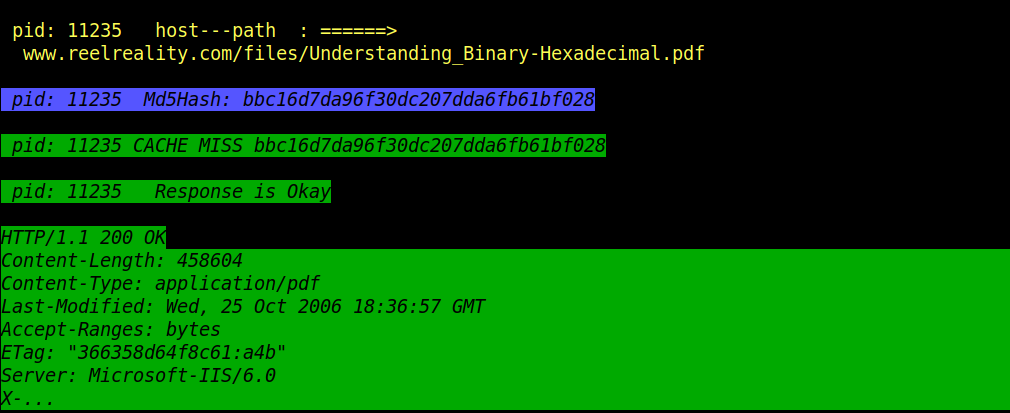




همین فایل در شاخه مربوطه تولید شده است.

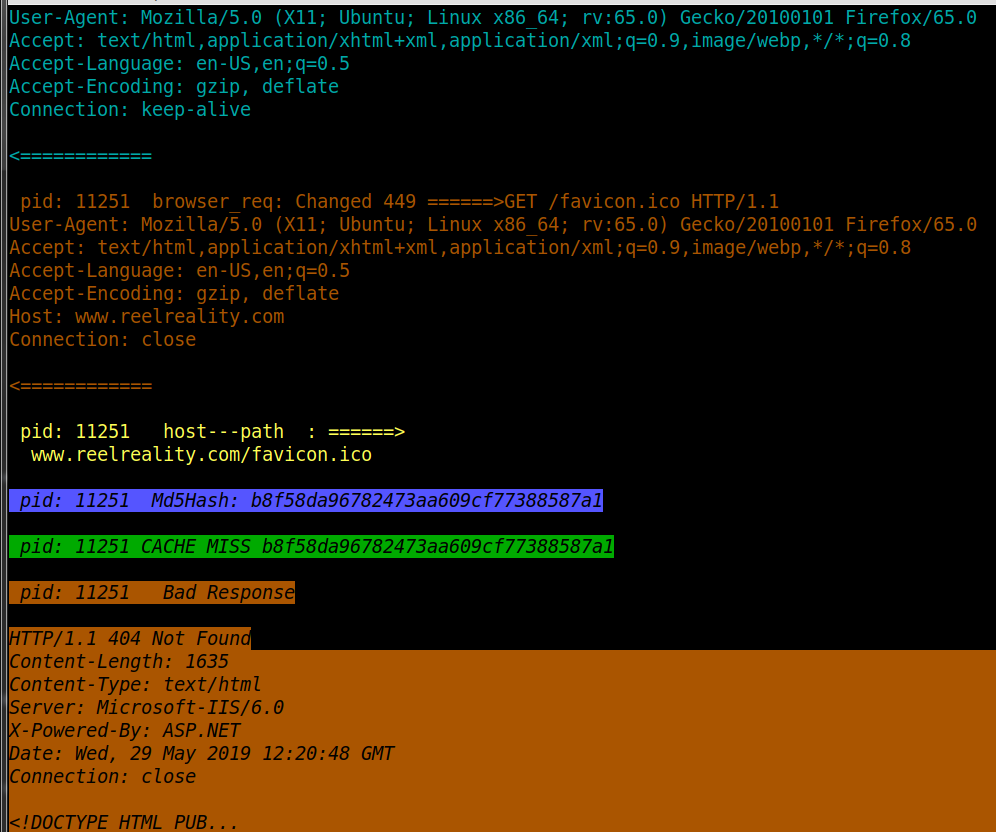


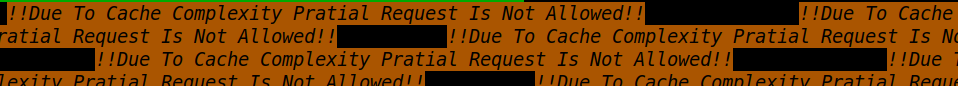
نمونه دوم درخواست فایل پی دی اف زیر و کنسل کردن درخواست در مرورگر





نمونه درخواست‌های ناموجود:



نمونه 

در مورد تحلیل استراتژی کش کردن حالت‌های متفاوتی ممکن است رخ دهد برای مثال فرض کنید سیاست Least Recently use استفاده شده و حجم حافظه کش 5 فایل باشد و این 5 فایل 100 بار بارگذاری شده باشند ، حال سایت دیگری را باز کرده و آن نیاز به 5 فایل دیگر دارد و آن ها در کش موجود نباشند، در نتیجه دائماً همین فایل‌های فعلی از کش اخراج می‌شوند (حتی اگر 1 عدد هم بیشتر نگه داریم همین وضعیت است)و درمورد FIFO نیز همچنان ممکن است MissRate با افزایش تعداد فایل ها بیشتر شود

و ..

\* تعدادی دستور مربوط به اجرای فایل کامپایل شده بر روی اوبونتو 18.04 توسط داکر نیز در انتها وجود دارد . که مرتبط با فضای کاری داخل ایمیج، کپی کردن فایل‌ها بر روی ایمیج و در معرض قرار دادن پورت بین ایمیج و محیط کاری و اجرای برنامه پروکسی .. هستند

test sites:

http://www.columbia.edu/~fdc/sample.html

http://rebeca-lang.org/

http://www.reelreality.com/files/Understanding\_Binary-Hexadecimal.pdf