

باسمه تعالی



گزارش طرح انتخابی VPN

رمزنگاری پیشرفته

دکتر سلماسی زاده

اعضای گروه:

کیخسرو خسروانی

پوریا دادخواه

ابوالفضل یوسفی

طرح انتخابی:

OpenVPN

در این گزارش به بررسی OpenVPN و پیشنهاد روشی برای احراز اصالت و دستیابی به اتصال ایمن در این VPN می پردازیم.

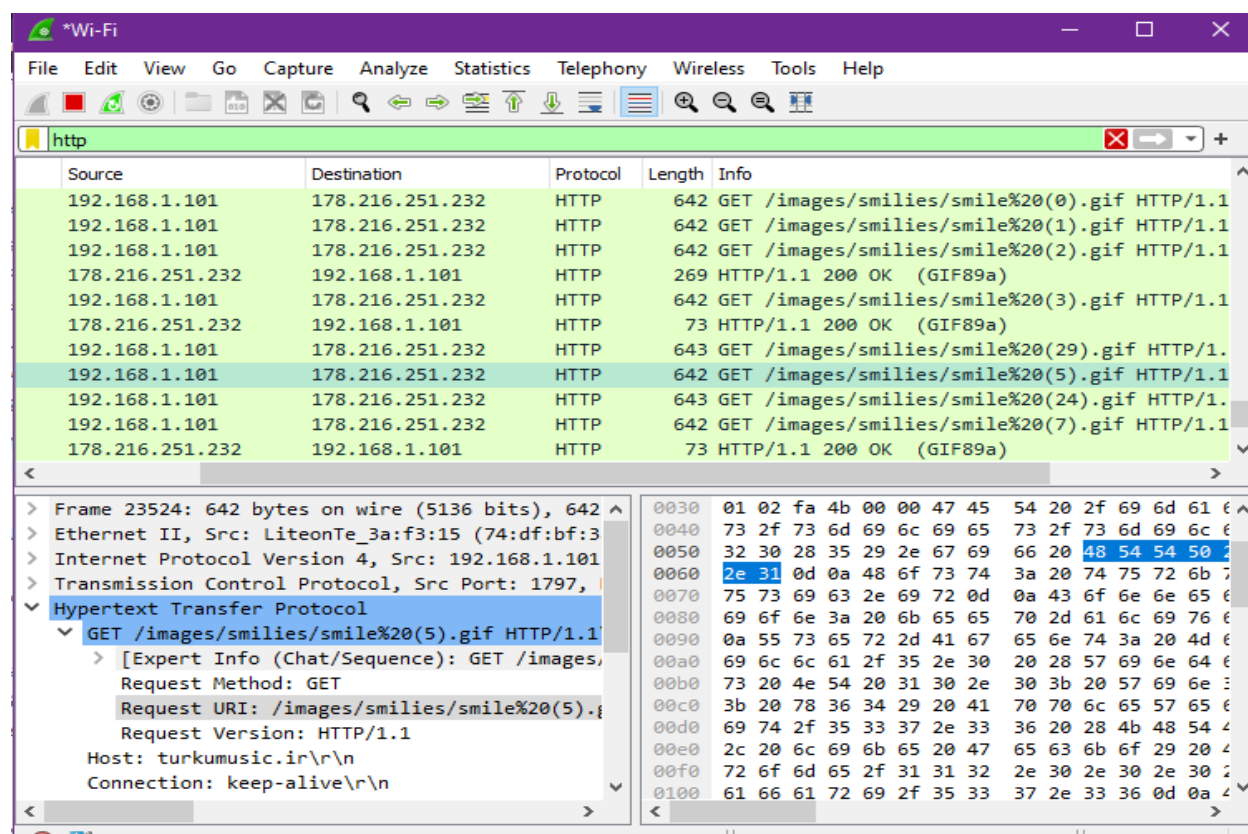
این گزارش به سه قسمت تقسیم بندی شده است :

- بخش اول : VPN چیست؟
در این بخش به معرفی VPN و کلیات آن، نحوه کارکرد آن و دسته بندی های VPN می پردازیم.
- بخش دوم : OpenVPN چگونه کار می کند؟
در این بخش به بررسی دقیق تر OVPN می پردازیم.
- بخش سوم: راهکارهای اصلاح و بهبود OpenVPN
در این بخش به ارائه روش هایی برای رسیدن به اتصال امن و احراز اصالت بسته های متعلق به OVPN از جانب سرور و نحوه رسیدن به کلید جلسه می پردازیم.
- بخش چهارم: جمع بندی
- بخش پنجم: مراجع

بخش اول: VPN چیست؟

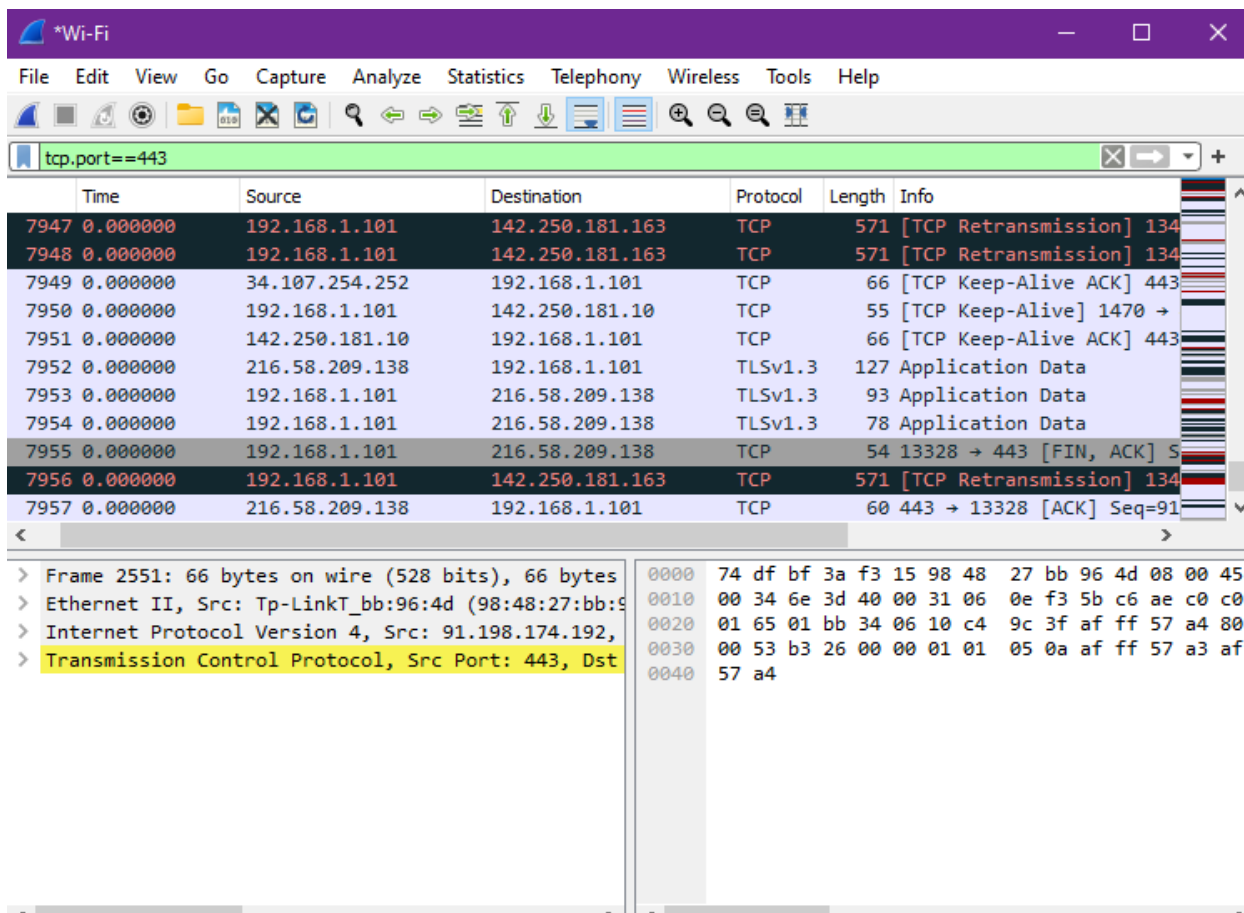
Virtual private network یا به اختصار VPN سرویسی‌هایی هستند که انتقال امن داده بین یک دستگاه و یک سرور را میسر می‌سازند. به زبان خیلی ساده یک VPN به کاربر اجازه می‌دهد یک شبکه محلی بین چند دستگاه در مکان‌های جغرافیایی مختلف ایجاد کند.

در حالت عادی بسته‌های ارسالی در بستر اینترنت در بسیاری از پروتکل‌ها مانند HTTP رمزنگاری نمی‌شوند و یک نفر سوم با دیدن آن بسته‌ها از محتوای آن و مبدأ و مقصد آن‌ها مطلع می‌شود. به عنوان مثال می‌توان شکل ۱ را که هنگام بازدید از یک سایت با پروتکل HTTP به وسیله Wireshark ضبط شده است را مشاهده کرد. همانطور که در شکل ۱ مشاهده می‌شود تمام پیام‌ها حتی عکس‌های مشاهده شده، قابل بازیابی از طریق Wireshark هستند.



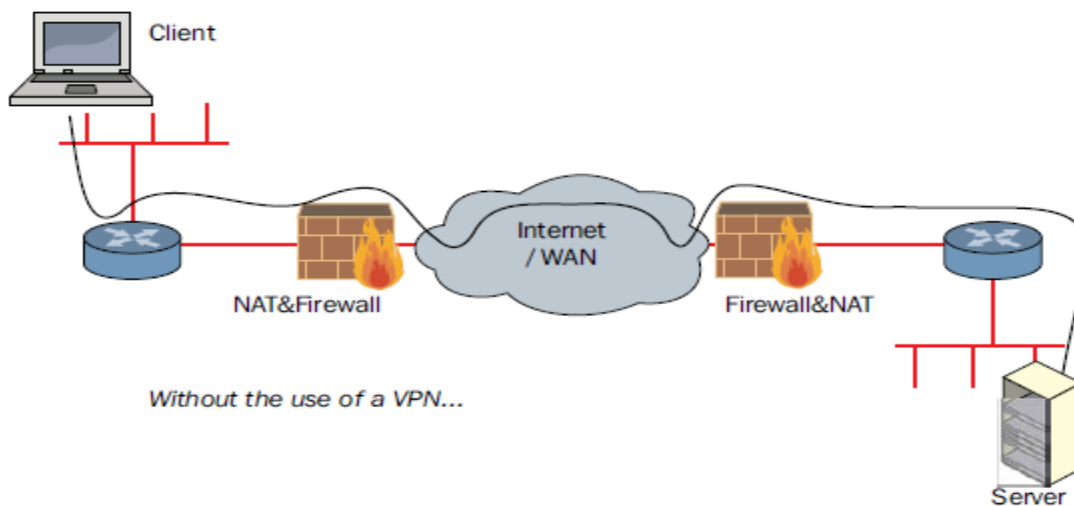
شکل ۱ - تصویر نرم‌افزار Wireshark در هنگام استفاده از پروتکل HTTP

حتی پروتکل‌هایی که ایمن‌تر هستند و از رمزنگاری استفاده می‌کنند مثل HTTPS همچنان با دیدن و دنبال کردن بسته‌ها می‌توان از مبدأ و مقصد و رفتار کاربر آگاه شد. برای مثال می‌توان شکل ۲ را مشاهده کرد که نشان‌دهنده بسته‌های پروتکل HTTPS است و در واقع از SSL استفاده می‌کند و بر روی پورت ۴۴۳ است. دیده می‌شود داده‌ها رمز شده و قابل مشاهده نیستند اما همچنان مبدأ و مقصد معلوم است.



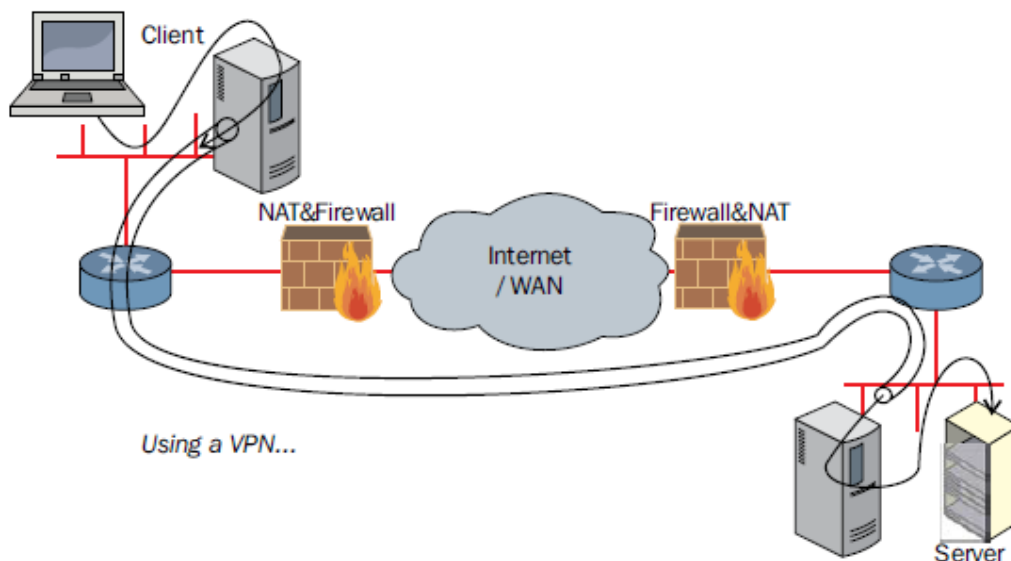
شکل ۲ - تصویر نرم‌افزار Wireshark در هنگام تحلیل بسته‌های HTTPS

در واقع VPN با رمزگذاری بسته‌ها باعث می‌شود که نفر سومی که کلید رمزگذاری را ندارد متوجه داده‌های ارسالی نشود. یک شبکه عادی بدون VPN مانند شکل ۳ عمل می‌کند.



شکل ۳ - نمایی از شبکه عادی بدون استفاده از VPN

در این مدل کاربر برای اتصال به سرور، داده‌های خود را به طور مستقیم روی شبکه ارسال می‌کند. از آنجا که این بسته‌ها از دو Firewall متعلق به ISP کاربر و سرور عبور می‌کند آنها در صورت تمایل می‌توانند مانع ارسال بسته شوند که این اساس عملکرد فیلترینگ است. حال اگر بخواهیم شبکه‌ای دارای VPN را نشان دهیم باید به شکل ۴ اشاره کنیم.



شکل ۴ - نمای شبکه دارای VPN

در این مدل، کاربر داده‌های خود را به صورت رمز شده به سرور VPN می‌فرستد و آن سرور، انتقال داده را انجام می‌دهد به عنوان مثال اگر دستور کاربر

`GET /eset_upd/consumer/latest/dll/update.ver.signed HTTP/1.1`

باشد به جای اتصال مستقیم به سرور eset این دستور را رمز شده به سرور VPN می‌فرستد و آن سرور به eset اتصال پیدا کرده و جواب را برای کاربر می‌فرستد.

پروتکل‌های VPN مختلفی از جمله PPTP , IPSEC , L2TP , OVPN وجود دارد که هر کدام به روش خود عمل احراز هویت و انتقال کلید جلسه را انجام می‌دهند اما هدف محافظت از بسته‌های کاربر در فضای اینترنت از طریق رمزنگاری است.

در زیر مراحل متداول مربوط به اتصال دستگاه به سرور VPN وجود دارد:

۱. کاربر از طریق دستگاه خود اتصال به سرور VPN را آغاز می‌کند.
۲. سرویس گیرنده VPN در دستگاه کاربر درخواستی را به سرور VPN ارسال می‌کند تا یک اتصال رمزگذاری شده برقرار کند.
۳. سرور VPN احراز اصالت کاربر را تأیید می‌کند و فرآیند رمزگذاری را آغاز می‌کند.
۴. هنگامی که فرآیند رمزگذاری کامل شد، سرور VPN از پروتکل VPN برای ایجاد یک تونل امن برای ترافیک اینترنت کاربر استفاده می‌کند.
۵. تمام داده‌های کاربر که از طریق این تونل امن عبور می‌کند، رمزگذاری شده و از تداخل خارجی محافظت می‌شود.
۶. دستگاه کاربر درخواست‌ها را به سرور VPN ارسال می‌کند و سرور آن درخواست‌ها را به اینترنت ارسال می‌کند، در حالی که آدرس IP کاربر و سایر اطلاعات شناسایی را پنهان می‌کند.

۷. سرور VPN پاسخ را از اینترنت دریافت می‌کند، آن را رمزگذاری می‌کند و از طریق تونل امن به دستگاه کاربر می‌فرستد.

۸. در نهایت، سرویس گیرنده VPN در دستگاه کاربر، داده‌ها را رمزگشایی کرده و در قالبی قابل استفاده به کاربر ارائه می‌دهد.

به طور خلاصه، پروتکل‌های VPN با ایجاد یک تونل امن و رمزگذاری شده بین کاربر و یک سرور کار می‌کنند که ارتباط امن را امکان‌پذیر می‌کند و از داده‌های کاربر در برابر تداخل خارجی محافظت می‌کند. پروتکل‌های VPN با هم اندکی تفاوت دارند، اما فرآیند اصلی رمزگذاری و ارسال ترافیک یکسان است.

معمولاً از VPN در موارد زیر استفاده می‌شود:

۱- استفاده از آن در شبکه‌های فاقد امنیت مانند Open Wi-Fi که در مکان‌های عمومی قرار دارند. با کمک VPN کسی نمی‌تواند محتوای مورد استفاده کاربر در اینترنت را متوجه شود.

۲- استفاده از VPN برای دور زدن سانسورهایی که دولت و ISP بر آنها تحمیل می‌کند.

۳- تغییر موقعیت جغرافیایی برای دسترسی به محتوایی که از طرف ارائه‌دهنده سرویس محدود به کشورهای خاصی شده است.

۴- ایجاد شبکه خصوصی داخلی توسط شرکت‌ها برای کارمندان خود

۵- استفاده از VPN برای امنیت بیشتر داده‌ها مثلاً در شبکه‌های ATM

اکثر پیاده‌سازی‌های VPN از نوعی رمزگذاری و علاوه بر آن، احراز هویت استفاده می‌کنند. رمزگذاری VPN تضمین می‌کند که هیچ نفر سومی که ممکن است ترافیک بین سیستم‌ها را نظارت کنند، نمی‌توانند داده‌های کاربر را رمزگشایی و تجزیه و تحلیل کنند.

احراز هویت دو جزء دارد که هر کدام در زمینه متفاوتی قرار دارند. جز اول، احراز هویت کاربر یا سیستم وجود دارد که تضمین می‌کند کسانی که به سرویس متصل می‌شوند مجاز هستند. این نوع احراز هویت ممکن است به صورت گواهی‌های هر کاربر یا ترکیبی از نام کاربری/رمز عبور باشد. علاوه بر این، قوانین خاص برای یک کاربر خاص مانند مسیرهای خاص، قوانین فایروال یا سایر اسکریپت‌ها و ابزارها پیاده‌سازی و تنظیم هستند.

دومین جزء احراز هویت، حفاظت از جریان ارتباطی است. در این مورد، روشی برای امضای هر بسته ارسال شده ایجاد می‌شود. هر سیستم قبل از رمزگشایی محموله تأیید می‌کند که بسته‌های VPN دریافتی به درستی امضا شده‌اند. با احراز هویت بسته‌هایی که قبلاً رمزگذاری شده‌اند، یک سیستم می‌تواند با حذف بسته‌هایی که قوانین احراز هویت را ندارند، در زمان پردازش صرفه‌جویی کند. در پایان، این امر از حمله (DoS) و Man in the Middle (MITM) با فرض ایمن نگه داشتن کلیدهای امضا جلوگیری می‌کند.

دسته بندی VPN ها :

همانطور که پیشتر اشاره شد VPN ها را می توان به ۴ مورد دسته بندی کرد:

۱- PPTP

۲- IPSec

۳- SSL

۴- OpenVPN

از آنجا که گزارش ما در مورد OVPN هست پس به بررسی آن می پردازیم.

OpenVPN :

OpenVPN اغلب یک VPN مبتنی بر SSL نامیده می شود، زیرا از پروتکل SSL/TLS برای ایمن سازی اتصال استفاده می کند. با این حال، OpenVPN همچنین از HMAC در ترکیب با یک الگوریتم هش برای اطمینان از یکپارچگی بسته های تحویل داده شده استفاده می کند. ویژگی معمولاً توسط سایر VPN های مبتنی بر SSL ارائه نمی شوند.

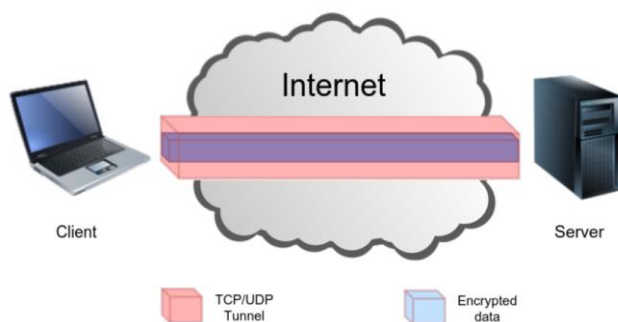
همچنین VPN های SSL بر مبنای وب هستند یعنی نیاز به برنامه و واسط کاربردی طرف کاربر نیستند در حالی که OpenVPN نیاز به واسط کاربری دارد. علاوه بر این، OpenVPN از یک آداپتور شبکه مجازی (یک دستگاه تن یا ضربه زدن) به عنوان رابط بین نرم افزار OpenVPN سطح کاربر و سیستم عامل استفاده می کند. به طور کلی، هر سیستم عاملی که از دستگاه tun/tap پشتیبانی می کند، می تواند OpenVPN را اجرا کند. این در حال حاضر شامل Linux، Free/Open/NetBSD، Solaris، AIX، Windows و Mac OS و همچنین دستگاه های iOS/Android می شود. برای همه این پلتفرم ها، نرم افزار کاربر باید نصب شود، که OpenVPN را از VPN های بدون کلاینت یا مبتنی بر وب متمایز می کند.

OpenVPN مفهوم یک کانال کنترل و یک کانال داده را دارد که هر دو به طور متفاوتی رمزگذاری و امن شده اند. با این حال، تمام ترافیک از طریق یک اتصال UDP یا TCP عبور می کند. کانال کنترل با استفاده از SSL/TLS رمزگذاری و ایمن می شود و کانال داده با استفاده از یک پروتکل رمزگذاری سفارشی رمز می شود. پروتکل و پورت پیش فرض برای OpenVPN، UDP و پورت ۱۱۹۴ است.

بخش دوم: پروتکل OpenVPN

• معرفی

OpenVPN یک SSL VPN است که لایه‌های ۲ و ۳ OSI شبکه ایمن را پیاده‌سازی می‌کند. این اجازه می‌دهد تا هر زیرشبکه IP روی یک پورت UDP یا TCP تونل شود و کاملاً به امنیت OpenSSL متکی است. ارتباط سطح بالا کلاینت و سرور در شکل ۵ نشان داده شده است. درست مانند سایر VPN ها، OpenVPN خدمات امنیتی ضروری مانند احراز هویت، رمزگذاری، حفاظت از یکپارچگی و کنترل دسترسی را ارائه می‌دهد.

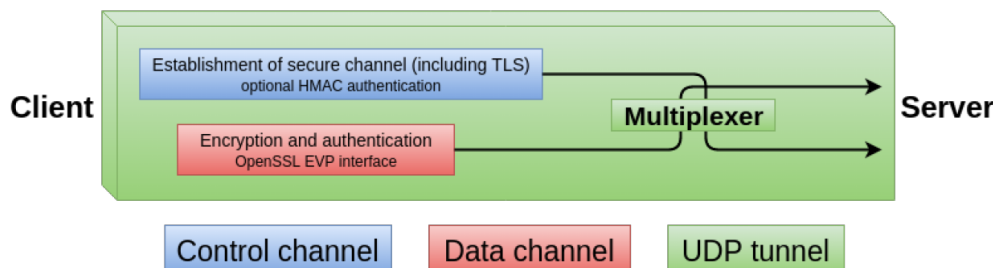


شکل ۵ - شمای ارتباط کلاینت و سرور در OpenVPN

OpenVPN از کتابخانه OpenSSL استفاده می‌کند که خود TLS را پیاده‌سازی می‌کند. TLS پروتکلی است که برای ایجاد یک اتصال ایمن از طریق یک شبکه ناامن طراحی شده است. TLS از گواهی‌های دیجیتال X.509 با رمزنگاری نامتقارن برای احراز هویت طرف مقابل و مذاکره در مورد کلید جلسه تونل متقارن استفاده می‌کند و از زیرپروتکل های Handshake، ChangeCipherSpec و Alert برای ایجاد یک اتصال ایمن استفاده می‌کند.

OpenVPN به کاربر این فرصت را می‌دهد تا از یک کلید ایستا (یا عبارت عبور از پیش به اشتراک گذاشته‌شده) برای تولید HMAC firewall استفاده کند که کل توالی Handshake TLS را تأیید می‌کند.

OpenVPN از دو کانال کنترل و داده استفاده می‌کند و جلسه TLS مورد استفاده برای احراز هویت و تبادل کلید (کانال کنترل) را با جریان داده‌های تونل رمزگذاری واقعی (کانال داده) چندگانه می‌کند.

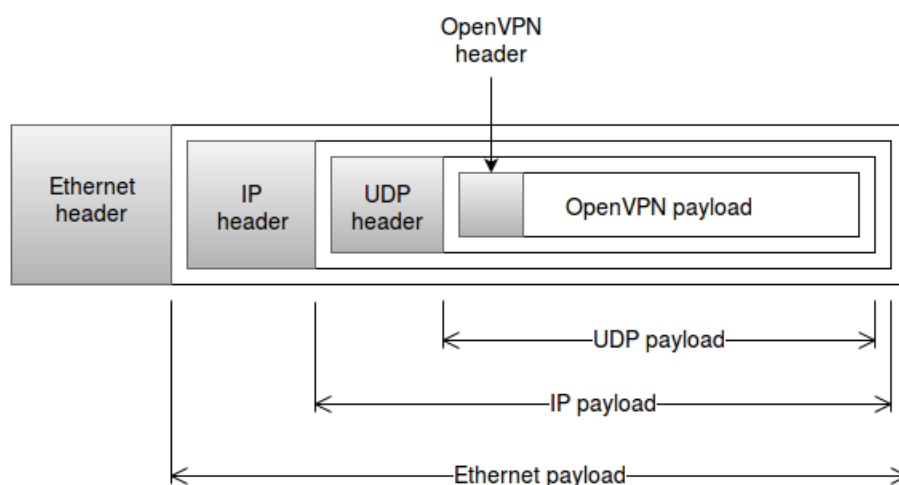


شکل ۶ - شمای جریان شبکه بین کلاینت و سرور در OpenVPN

از آنجایی که UDP یک پروتکل بدون اتصال است، بسته‌های IP رمزگذاری شده و امضا شده روی UDP بدون هیچ گونه تضمین قابلیت اطمینان تونل می‌شوند. قابلیت اطمینان مورد نیاز برای احراز هویت ایمن توسط پروتکل TLS ارائه می‌شود که از TCP به

عنوان لایه قابلیت اطمینان خود استفاده می‌کند. توجه می‌کنیم که هر دو کانال داده و کنترل علی‌رغم کارکرد متفاوت در یک تونل یکسان هستند.

OpenVPN گزینه‌های مختلفی را برای پی‌گیری و برقراری اتصال VPN فراهم می‌کند. ساختار کلی و پیش‌فرض بسته OpenVPN و کپسوله‌سازی متعاقب آن را می‌توان در شکل ۷ مشاهده کرد و بسته‌های متفاوت payloadهای متفاوت دارند. (مانند داده‌های کپسوله شده TLS)



شکل ۷ - کپسوله‌سازی بسته‌های OpenVPN

با توجه به انعطاف‌پذیر پروتکل از مودهای متفاوتی برای احراز هویت و پیاده‌سازی TLS می‌توان استفاده کرد که در ادامه به توضیح خلاصه آن می‌پردازیم.

• روش‌های احراز هویت

OpenVPN دو حالت احراز هویت متفاوت را ارائه می‌دهد. حالت اصلی از TLS با گواهی‌نامه‌ها و/یا اعتبار کاربر برای تأمین امنیت استفاده می‌کند. حالت دیگر از کلیدهای ثابت از پیش مشترک استفاده می‌کند که به طور پیش‌فرض احراز هویت و رمزگذاری را فراهم می‌کند. چندین تفاوت و ملاحظات امنیتی برای استفاده از هر یک از حالت‌ها وجود دارد. مقایسه سطح بالا را می‌توان در جدول ۱ مشاهده کرد.

	Pre-shared static key mode	TLS mode
Crypto mode	Symmetric	Asymmetric & Symmetric
Handshake	No	Yes
Implementation	Simpler	More complex
Speed	Faster	Slower
CPU consumption	Smaller	Higher
Scalability	Limited	Very good
Key exchange	No	Yes
Encryption keys renewal	No	Yes
Peer authentication	Yes	Yes
Perfect forward secrecy	No	Yes

جدول ۱ - مقایسه دو روش احراز هویت عنوان شده

هر دو حالت کلید ایستا از قبل به اشتراک گذاشته شده و حالت TLS امنیت معادل را با فرض اینکه به درستی پیاده سازی شده‌اند و از رمزهای اولیه رمزنگاری یکسان استفاده می‌کنند، فراهم می‌کنند. با این حال، حالت کلید استاتیک از قبل به اشتراک گذاشته شده مکانیسم تجدید کلید را ارائه نمی‌دهد و بنابراین رازداری کامل به جلو ارائه نمی‌شود. این می‌تواند به عنوان یک نگرانی امنیتی در نظر گرفته شود. از آنجایی که حالت کلید استاتیک از پیش به اشتراک گذاشته شده از الگوریتم کلید متقارن استفاده می‌کند، پیاده‌سازی آن ساده‌تر است و عملکرد آن در عملیات رمزنگاری بهبود یافته است. با این حال، حالت کلید استاتیک از قبل به اشتراک گذاشته شده، مقیاس‌پذیری بسیار محدودی دارد، زیرا امکان تعویض خودکار کلید وجود ندارد. علاوه بر این، کلید متقارن باید در هر همتای OpenVPN به صورت متن ساده وجود داشته باشد که یک چالش امنیتی اضافی است. احراز هویت همتا در حالت کلید استاتیک از قبل به اشتراک گذاشته شده تنها با مالکیت کلید متقارن مشترک متعلق به هر یک از همتایان ارائه می‌شود.

• ارتباط کلاینت-سرور و پیام‌ها

این بخش فقط به پیام‌های حالت TLS و احراز هویت با استفاده از تنظیمات پیش فرض OpenVPN همانطور که توضیح داده شده است مربوط می‌شود. حالت کلید ایستا از پیش مشترک مکانیسم‌های تبادل کلید و احراز هویت را ندارد، بنابراین ارتباط کلاینت-سرور حالت بعدی مورد بحث قرار نمی‌گیرد.

حالت OpenVPN TLS دارای هشت نوع پیام مختلف است که برای ایجاد (کانال کنترل) و مدیریت (کانال داده) تونل VPN استفاده می‌شود. ساختار کلی ارتباط کلاینت با سرور را از شروع ساختن تونل بر بستر TLS توسط پیام‌های کنترلی تا تبادل داده‌ها در تونل بین این دو و حفظ و تمدید کلید برای برقراری ارتباط را در شکل ۸ می‌توان مشاهده و خلاصه کرد که توضیح مراحل آن را در دو بخش انجام می‌دهیم:

کانال کنترلی:

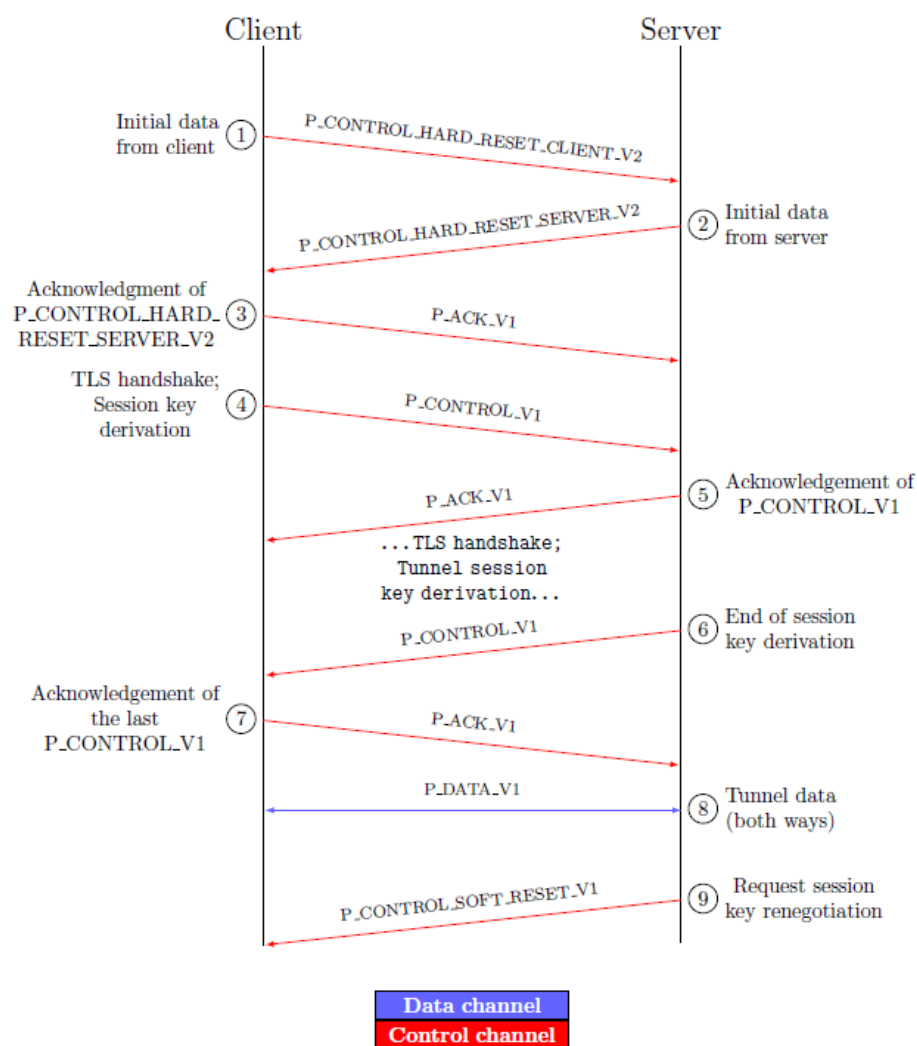
مرحله ۱، ۲ و ۳: بسته‌های اولیه‌سازی جلسه بین کلاینت و سرور به همراه بسته‌های تأیید مربوطه ارسال می‌شوند.

مرحله ۴، ۵، ۶ و ۷: دست دادن TLS درون لایه پروتکل OpenVPN کپسوله شده است. پس از دست دادن موفقیت آمیز TLS، استخراج کلید جلسه تونل انجام می‌شود.

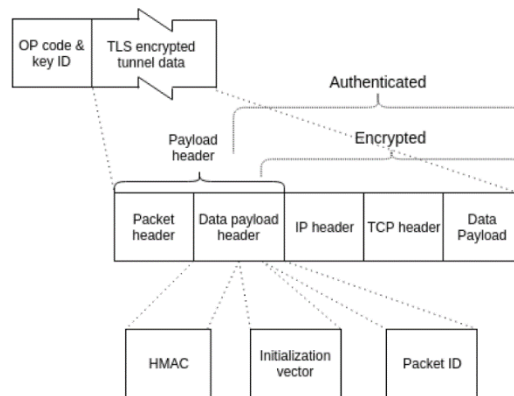
مرحله ۹: درخواست مذاکره مجدد کلید جلسه تونل.

کانال داده:

مرحله ۸ بوده که مشتری و سرور در یک بستر UDP بایکدیگر صحبت می کنند که ساختار عمومی بسته های داده در شکل بعدی آورده شده است.



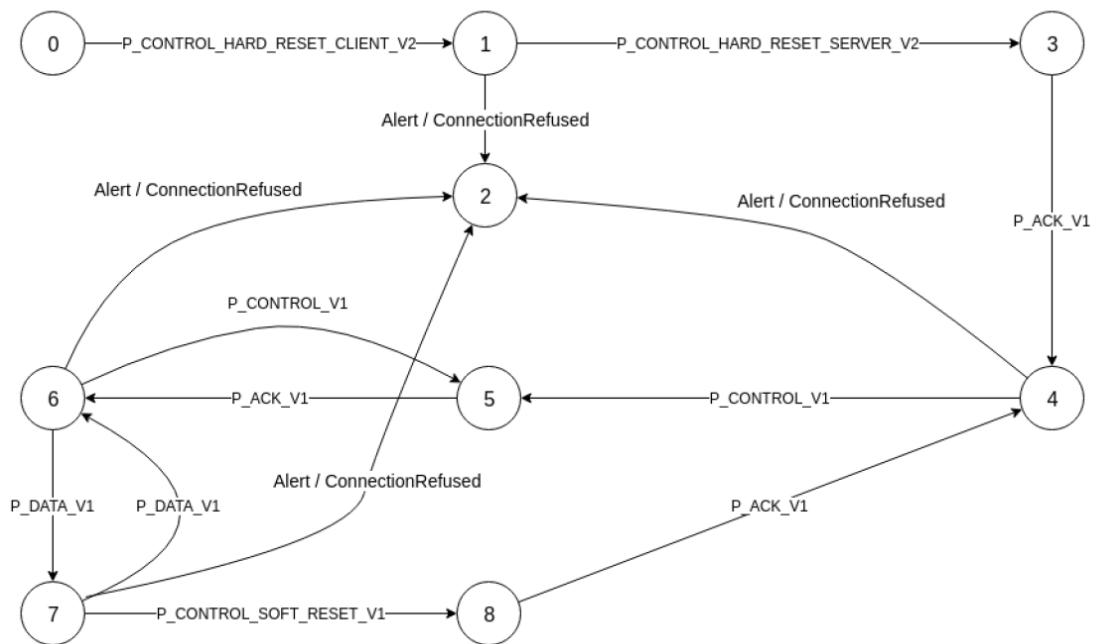
شکل ۸ - ارتباط بین کلاینت و سرور



شکل ۹ - ساختار بسته داده

• ماشین حالت OpenVPN

در نهایت با استفاده از دانش اتوماتا، یک ماشین حالت محدود برای OpenVPN طراحی می‌کنیم که نحوه عملکرد دقیق آن را متوجه شویم و در مواقع رخداد خطا به درستی واکنش مناسب را ارائه دهیم. این ماشین از همان پروسه معرفی شده در بخش قبل پیروی می‌کند:



شکل ۱۰ - ماشین حالت OpenVPN

انتقال ۱ <- ۲ <- ۳ <- ۴ و ۴ <- ۷ <- ۸ <- ۴ به کانال کنترلی اشاره دارد که برای ایجاد کانال امن VPN بین دو طرف استفاده می‌شود. کانال کنترل شامل دست دادن TLS است که در انتقال های زیر انجام می‌شود: ۴ <- ۵ <- ۶. انتقال داده در تونل امن به صورت زیر عمل می‌کند: ۶ <- ۷.

تعداد انتقال های تکراری به اندازه بسته و همچنین مقدار داده ای که منتقل می شود بستگی دارد.

در آخر باید اشاره کرد که همان طور که گفته شد ساختار عمومی پروتکل مورد بررسی قرار گرفت و باتوجه به قابلیت هایی که ساختار پروتکل در اختیار ما می گذارد می توان ویژگی های دیگری بر روی آن پیاده کرد که امنیت، کارایی و یا موارد مورد انتظار خود را تقویت کنیم. مثلا با استفاده از HMAC Firewall که به آن اشاره شد می توان از Buffer Overflow و DoS attacks جلوگیری کرد.

بخش سوم: ارائه راهکارهای اصلاح و بهبود پروتکل OpenVPN

حال با توجه به نکات گفته شده دو راهکار برای اصلاح پروتکل OpenVPN و دور زدن فیلترینگ ارائه داده شده است که به شرح ذیل هستند:

۱- تغییر در handshake پروتکل OpenVPN

۲- کپسوله سازی پروتکل OpenVPN با استفاده از تکنیک‌های مبهم‌سازی (obfuscation)

• تغییر در handshake پروتکل OpenVPN

راهکار دیگر رفع مشکلی است که باعث شناسایی و در نتیجه بلاک کردن بسته‌های OpenVPN می‌شود و آن handshake اولیه‌ای است که در ابتدای پروتکل برای رسیدن به کلید جلسه برقرار می‌شود که برای حل آن دو پیشنهاد ارائه می‌دهیم:

الف) پروتکل برقراری کلید OpenVPN را به پروتکل دیگری مانند IKE v2 عوض کنیم.

ب) کاری کنیم که حتی handshake اولیه که هنوز سر کلید رمزنگاری توافق نکرده‌اند نیز رمز شده باشد.

الف) تغییر پروتکل برقراری کلید جلسه:

پروتکلی که در OpenVPN به عنوان کلید جلسه استفاده می‌شود در شکل ۱۱ نشان داده شده است. که مراحل آن به صورت زیر است:

۱) کاربر درخواست برقراری ارتباط می‌دهد، یک نانس می‌فرستد و نسخه پروتکلی که پشتیبانی می‌کند و روش رمزنگاری را ارسال می‌کند.

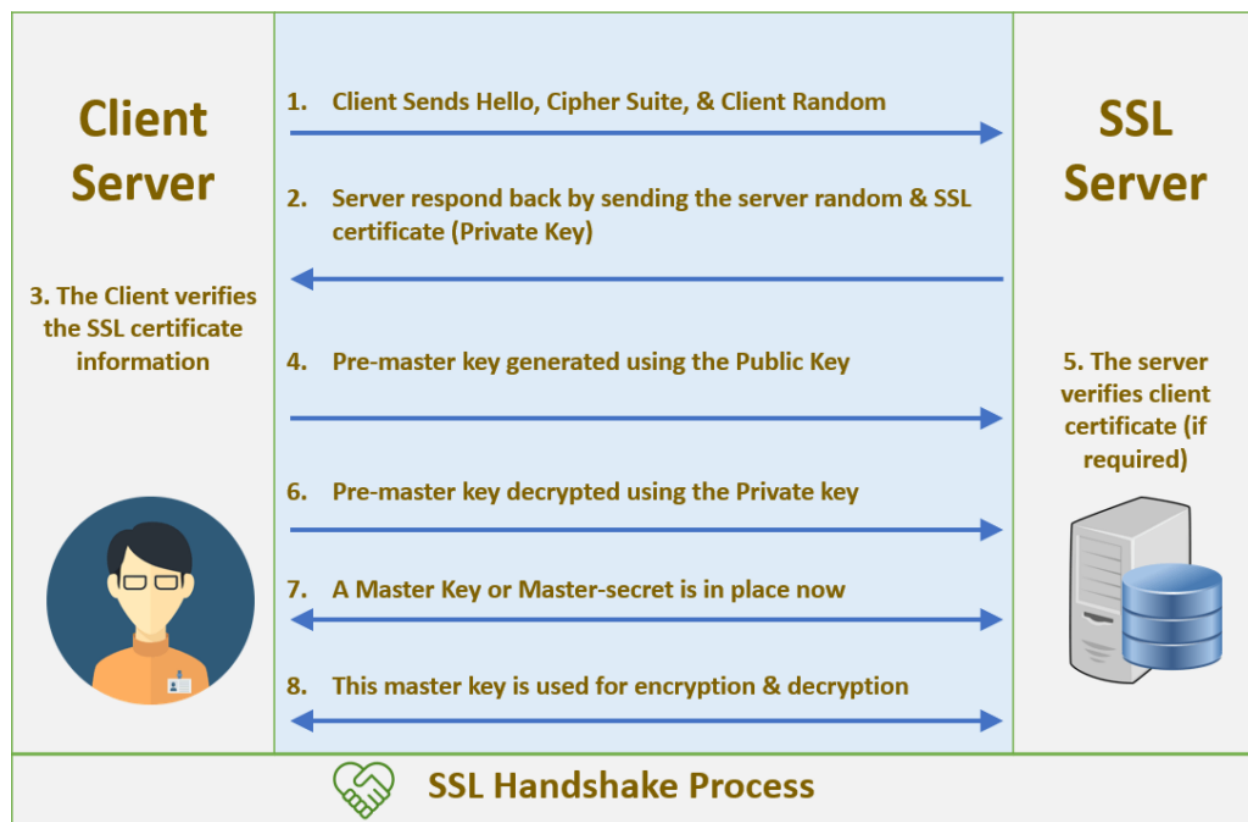
۲) سرور در جواب یک Ack به پیام کاربر ارسال می‌کند و همچنین گواهی معتبر خود را همراه با نانس خود و نانس مرحله ۱ کاربر می‌فرستد.

۳) کاربر معتبر بودن گواهی را بررسی می‌کند و در صورت تایید یک کلید جلسه تولید و با کلید عمومی سرور که در گواهی بود رمز می‌کند.

۴) سرور در صورت نیاز جواز کاربر را چک می‌کند (برای مدل دارای گواهی) و سپس پیام دریافتی را با کلید خصوصی خود باز می‌کند و در صورت یکی بودن با نانس خود از مرحله ۲، آن کلید را به عنوان کلید جلسه در نظر می‌گیرد و یک چالش را با آن رمز کرده و به کاربر می‌فرستد.

۵) کاربر پیام را دریافت می‌کند و اگر توانست با کلیدی که خود تولید کرده بود آن را باز کند نتیجه می‌گیرد که سرور کلید را می‌داند.

۶) سپس جواب چالش را با همان کلید رمز کرده و برای سرور می‌فرستد و سرور هم مطمئن می‌شود کاربر کلید را می‌داند و از آنجا به بعد آن کلید را به عنوان کلید جلسه در نظر می‌گیرند و رمزنگاری‌ها با آن کلید صورت می‌گیرد.



شکل ۱۱ - فرآیند handshake پروتکل SSL

این پروتکل مرحله hello و Ack و ارسال گواهی فاقد رمزنگاری هست و ISP می‌تواند آن را تشخیص بدهد. پس اگر هدف این است که بدون کپسوله کردن VPN را بهبود دهیم باید این مرحله که دلیل کشف شدن VPN هست را درست کنیم.

تغییر پروتکل برقراری کلید OpenVPN به پروتکل IKE v2 را Express VPN پیاده‌سازی کرده است و در ایران اتصال برقرار می‌شود.

OpenVPN از پروتکل TLS/SSL برای رسیدن به کلید جلسه استفاده می‌کند که ساختار آن با برقراری کلید و احراز اصالت در HTTPS متفاوت است و در نتیجه حتی اگر روی پورت ۴۴۳ قرار گیرد شناخته می‌شود. IKE v2 از پروتکل استاندارد RFC 7296 استفاده می‌کند و ایمن‌تر از TLS است.

ب) رمزنگاری پروتوکل برقراری کلید جلسه:

از نسخه OpenVPN v2.4 به بعد یک ویژگی به OpenVPN اضافه شده که آن TLS-crypt نام دارد. این مدل به این صورت است که اگر فعال باشد سرور هنگام ساختن فایل config برای client درون فایل یک رشته کلید هم قرار می‌دهد که آن pre-shared key بین سرور و client می‌شود. هر دو همواره زمانی که می‌خواهند پروتکل TLS را اجرا کنند تمام پیام‌هایشان را رمزنگاری متقارن با آن کلید مشترک می‌کنند. بعد از رسیدن به کلید جلسه باقی رمزنگاری‌ها را با آن کلید جلسه انجام می‌دهند. این کار دو مزیت دارد:

- حتی اتصال اولیه و فرستادن Ack و گواهی اولیه نیز تماماً رمز می‌شود پس ISP صرفاً یک ترافیک رمز شده روی شبکه می‌بیند و نمی‌تواند تشخیص دهد که یک TLS هست.
- نقش احراز اصالت هم دارد و این باعث می‌شود که حمله DoS به آن زده نشود. در حالت عادی یک مهاجم با دادن تعداد زیادی درخواست برقراری کلید می‌توانست حافظه و توان دستگاه را اشغال کند به این ترتیب که برای هر برقراری کلید سرور باید اطلاعاتی از قبیل IP و port تقاضا بدهد، نانس او و نانس تولیدی خود و ... را ذخیره نگه دارد و از آنجا که تا ۶۰ ثانیه همچنان معتبر هست این یعنی با دادن درخواست بالا می‌توان مانع اتصال بقیه به VPN شد (مانع اجرای handshake) اما در این مدل چون پیام‌ها رمز است کاربر غیرمجاز نمی‌تواند حمله DoS بزند زیرا کلید رمزنگاری را ندارد پس سرور خیلی سریع با رمزگشایی پیام و رسیدن به متن بی‌معنی اتصال را قطع و حافظه را آزاد می‌کند.

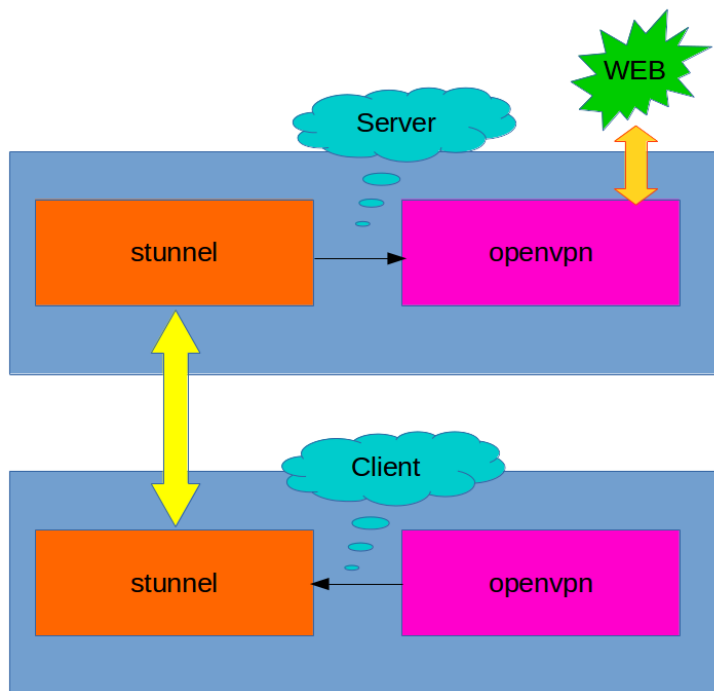
• کیسوله‌سازی پروتکل OpenVPN با استفاده از تکنیک‌های مبهم‌سازی (obfuscation)

ایده کلی این روش به این صورت است که به جای اینکه ارتباط کلاینت و سرور با پروتکل و ساختار بسته پیش فرض OpenVPN باشد که قابل تشخیص توسط DPI است، از طریق یک ابزار واسطه، بسته OpenVPN را داخل یک بسته با فرمت معمول شبکه مثل SSL/TLS قرار دهیم و دو طرف ارتباط پس از دریافت آن بسته، payload را باز کرده و با بسته اصلی OpenVPN کار کنند. برای این هدف دو روش قابل پیاده‌سازی معرفی می‌کنیم و یکی از این دو را کامل پیاده‌سازی کرده و با ابزارهای شبکه درستی کارکرد آن را نشان می‌دهیم.

1. Stunnel

همانطور که در شکل ۱۲ مشاهده می‌کنید، ترافیک بدون توجه به پروتکل داخلی آن توسط Stunnel به صورت SSL/TLS کیسوله می‌شود. از آنجایی که ما به SSL/TLS handshake نیاز داریم، اگر OpenVPN در پروتکل اصلی است، باید از پروتکل TCP برای OpenVPN استفاده کنیم.

در واقعیت، ترافیک SSL/TLS کوتاه و متناوب است، بنابراین هنوز هم برای یک دولت/ISP تشخیص Stunnel آسان است زیرا تعداد زیادی از ترافیک به عنوان SSL/TLS منتقل می‌شوند. توصیه می‌شود از پورت TCP ۴۴۳ یا TCP ۵۸۷ برای مخفی کردن ترافیک تاکنون استفاده کنید.



شکل ۱۲ - جریان ترافیک در Stunnel

2. Obfsproxy

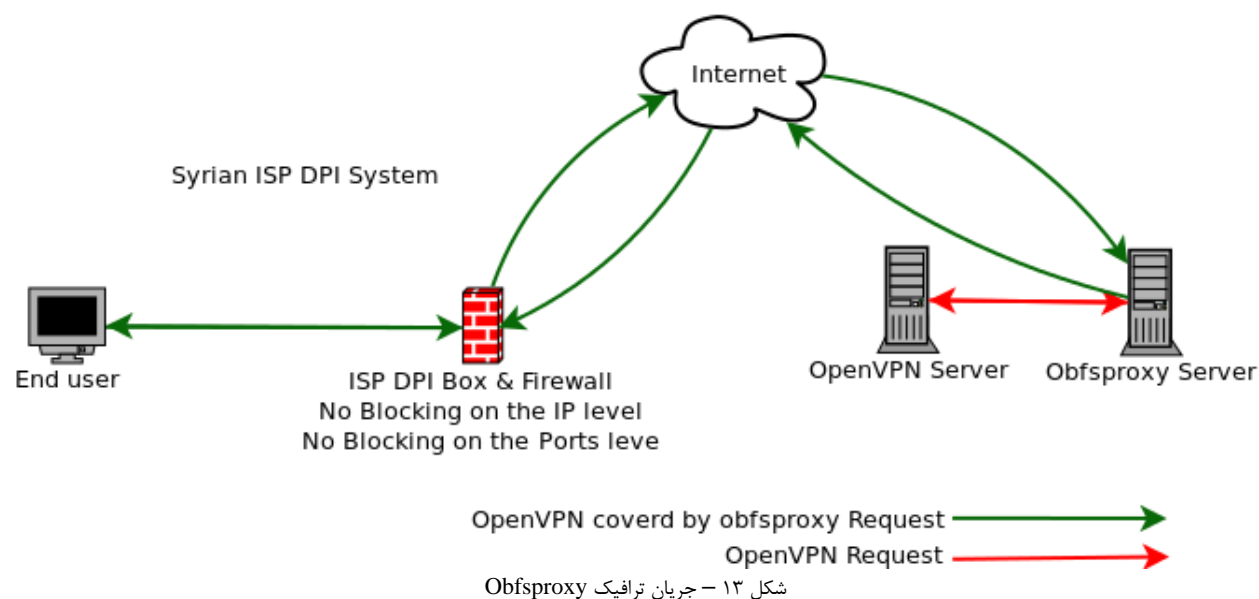
در ابتدا بهتر است به فرآیند مبهم‌سازی یا همان Obfuscation بپردازیم. منظور از مبهم‌سازی آن است که یک هویت را چنان بیان کنیم که معنی آن به راحتی قابل درک نباشد. از مبهم‌سازی در شبکه، برنامه‌نویسی، مخابرات و ... استفاده می‌گردد. هدف اصلی مبهم‌سازی در شبکه، پنهان‌سازی داده یا جریان شبکه از DPI است.

رویکردهای مبهم‌سازی در شبکه عبارتند از:

- رمزنگاری: در این رویکرد با رمز کردن داده‌ها، در محتوای هر بسته ابهام ایجاد می‌کنیم. ممکن است برخی الگوریتم‌های رمزنگاری ابرداده (metadata) خود را به قسمت داده‌ها یا هدر بسته اضافه نمایند. به همین دلیل برخی از پارامترهای بسته‌های رمزنگاری‌شده قابلیت الگویابی دارند و می‌توانند توسط DPI‌ها شناسایی شوند. در فیلترینگ کشورهای چین و ایران، بسته‌های رمز شده شناسایی می‌شوند.

- تصادفی‌سازی: منظور از تصادفی‌سازی، تصادفی نمایاندن بایت‌های بسته یا مسیریابی بسته است. در صورتی این رویکرد موفق واقع می‌گردد که سیاست لیست سیاه در ISP‌ها اجرا گردد.
- ایجاد تونل: تفاوتی که بین تونل و VPN و پروکسی وجود دارد آن است که VPN‌ها و پروکسی‌ها بعد از ایجاد ارتباط دارای ابر داده‌هایی هستند که نمایانگر خود هستند و بعضاً این ابزار ضدسانسور خود مورد شناسایی واقع می‌گردند. به همین دلیل تونل‌ها با استفاده از زیرساخت‌های آزاد وب بسته‌ها را به صورت ناشناس عبور می‌دهند و خود ابزاری برای مبهم‌سازی ابزارهای ضدسانسور هستند.
- تقلید: این رویکرد به این نحو است که سعی می‌شود بسته‌ها شبیه به بسته یک پروتکل معتبر به نظر برسند و برای ISP‌ها غیرواقعی به نظر نرسند. عملاً برای موفقیت در مواقعی بکار می‌رود که در ISP از سیاست لیست سفید استفاده می‌گردد.

در واقع Obfsproxy از رویکردهای تصادفی‌سازی و تقلید برای در امان ماندن از فیلترینگ استفاده می‌کند. نقش Obfsproxy این است که جریان ترافیک Tor را آسان کند تا به هر چیزی که دوست داریم شبیه شود. به این ترتیب Tor می‌تواند بر امنیت و ناشناس بودن تمرکز کند و Obfsproxy می‌تواند بر ظاهر تمرکز کند. ماژول “obfs3” یک پوشش رمزگذاری در اطراف ترافیک Tor اضافه می‌کند، با استفاده از یک دست دادن که هیچ الگوی بایستی قابل تشخیصی ندارد. به طور مشابه پیش‌نیاز این ابزار این است که OpenVPN را روی پروتکل TCP پیاده کرده باشیم.



در ادامه ابتدا Obfsproxy روی OpenVPN را در دو طرف ارتباط کلاینت و سرور پیاده کنیم و نتایج ارتباط را ببینیم. پس ابتدا روند نصب و کانفیگ کردن OpenVPN را بیان می‌کنیم و سپس Obfsproxy را روی آن‌ها نصب می‌کنیم. این پیاده‌سازی به دو روش شرح داده می‌شود که در اولین روش بین کامپیوترهای VPN، شبکه‌ای از نوع Nat برقرار است و در روش دوم شبکه‌ای از نوع Host-only برقرار است.

• روش اول

برای این کار ابتدا دو virtual machine با Ubuntu 18.04 tls راه‌اندازی کردیم. سپس شبکه بین VM‌ها را Nat network قرار دادیم و رنج IP آنها را 255.255.255.0 یا به عبارتی 10.0.3.x قرار دادیم. سپس یک از دو ماشین مجازی را به عنوان server با آدرس 10.0.3.5 و دیگری به عنوان client با آدرس 10.0.3.4 قرار دادیم. سپس OpenVPN را در هر دو نصب کردیم و در سمت client از دستور زیر استفاده کردیم:

```
sudo apt update && sudo apt install openvpn -y
```

و در سمت سرور از دستور زیر :

```
sudo curl -O https://raw.githubusercontent.com/angristan/openvpn-install/master/openvpn-install.sh
```

```
sudo chmod +x openvpn-install.sh
```

```
sudo ./openvpn-install.sh
```

برای کانفیگ کردن مراحل زیر را انجام دادیم:

```
Ip= 10.0.3.5 (1)
Ip=10.0.3.5 (2)
Port=443 (3)
Protocol = TCP (4)
Ipv6 no (5)
No custom enc (6)
Certificateless user (7)
Chose a username for user (8)
```

سپس فایل کانفیگی که سرور ساخت را به client's VM انتقال می‌دهیم. برای مرحله آخر باید Obfsproxy را روی سرور فعال کرده و به او مشخص کنیم که روی کدام پورت بخواند و حاصل را به کجا انتقال دهد.

برای نصب Obfsproxy کد زیر را قرار دادیم.

```
sudo apt-get install obfsproxy -y
```

سپس آن را روی پورت و IP درست تنظیم می کنیم:

```
obfsproxy obfs3 --dest=127.0.0.1:443 server 0.0.0:21194
```

که در اصل از پورت 443 خود سیستم که OpenVPN هست را به عنوان مقصد می گذارد و روی پورت 21194 برای تمام IP های مبدأ پکت ها را می خواند. حال باید در سمت client اعلام کنیم که ترافیک را روی پورت 21194 قرار دهد. اکنون کارها طرف سرور تمام شد.

در سمت client ابتدا OpenVPN را با دستور زیر نصب می کنیم:

```
sudo apt update && sudo apt install openvpn -y
```

سپس Obfsproxy را نصب می کنیم.

```
sudo apt-get install -y obfsproxy
```

سپس Obfsproxy را روی پورت و IP درست تنظیم می کنیم:

```
obfsproxy obfs3 socks 127.0.0.1:10194
```

که در واقع ترافیک پورت 10194 را می پوشاند که پورتهی هست که OpenVPN client روی آن کار می کند. سپس فایل کانفیگی که از server گرفته بودیم و به client's VM منتقل کردیم را باید یک تغییر بدهیم به این شکل که اول آن این دو خط را اضافه کنیم:

```
socks-proxy-retry
```

```
socks-proxy 127.0.0.1 10194
```

و پورت 443 در خط زیر را به 10194 تغییر دهیم:

```
remote 10.0.3.5 21194
```

فایل نهایی را ذخیره می کنیم و درخواست برقراری به VPN را به کمک دستور زیر و فایل conf که در بالا آن را درست کردیم اجرا می کنیم:

```
sudo openvpn --config ./username.ovpn
```

حال مطابق شکل ۱۴ اتصال به درستی برقرار می شود.

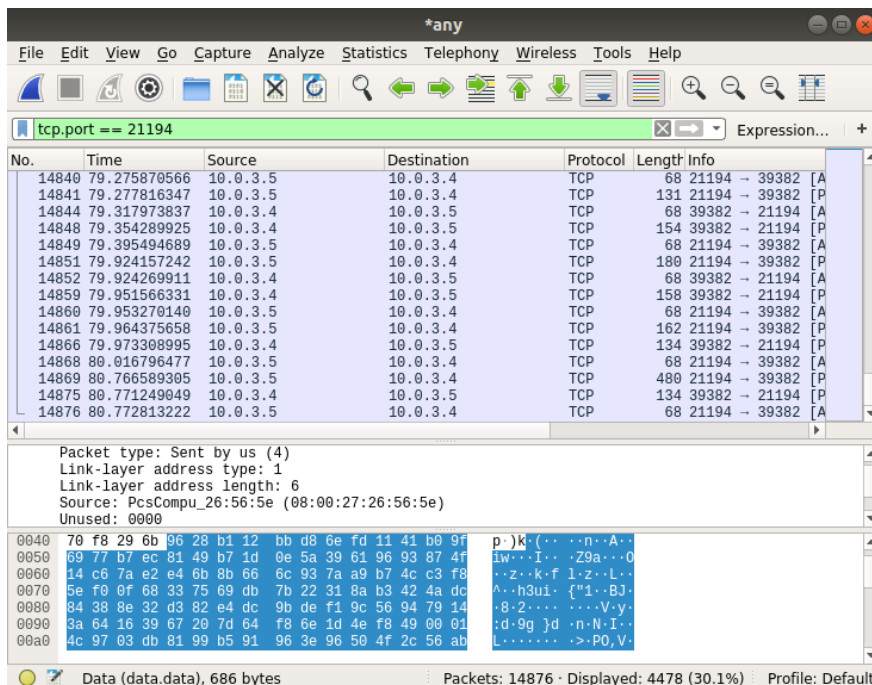
```

natclient@natclient: ~
File Edit View Search Terminal Help
dhcp,route-gateway 10.8.0.1,topology subnet,ping 10,ping-restart 120,ifconfig 1
0.8.0.2 255.255.255.0,peer-id 0,cipher AES-128-GCM'
Sun May 28 08:31:54 2023 OPTIONS IMPORT: timers and/or timeouts modified
Sun May 28 08:31:54 2023 OPTIONS IMPORT: --ifconfig/up options modified
Sun May 28 08:31:54 2023 OPTIONS IMPORT: route options modified
Sun May 28 08:31:54 2023 OPTIONS IMPORT: route-related options modified
Sun May 28 08:31:54 2023 OPTIONS IMPORT: --ip-win32 and/or --dhcp-option option
s modified
Sun May 28 08:31:54 2023 OPTIONS IMPORT: peer-id set
Sun May 28 08:31:54 2023 OPTIONS IMPORT: adjusting link_mtu to 1624
Sun May 28 08:31:54 2023 OPTIONS IMPORT: data channel crypto options modified
Sun May 28 08:31:54 2023 Outgoing Data Channel: Cipher 'AES-128-GCM' initialize
d with 128 bit key
Sun May 28 08:31:54 2023 Incoming Data Channel: Cipher 'AES-128-GCM' initialize
d with 128 bit key
Sun May 28 08:31:54 2023 ROUTE_GATEWAY 10.0.3.1/255.255.255.0 IFACE=enp0s3 HWAD
DR=08:00:27:26:56:5e
Sun May 28 08:31:54 2023 TUN/TAP device tun0 opened
Sun May 28 08:31:54 2023 TUN/TAP TX queue length set to 100
Sun May 28 08:31:54 2023 do_ifconfig, tt->did_ifconfig_ipv6_setup=0
Sun May 28 08:31:54 2023 /sbin/ip link set dev tun0 up mtu 1500
Sun May 28 08:31:55 2023 /sbin/ip addr add dev tun0 10.8.0.2/24 broadcast 10.8.
0.255
Sun May 28 08:31:55 2023 /sbin/ip route add 10.0.3.5/32 dev enp0s3
Sun May 28 08:31:55 2023 /sbin/ip route add 0.0.0.0/1 via 10.8.0.1
Sun May 28 08:31:55 2023 /sbin/ip route add 128.0.0.0/1 via 10.8.0.1
Sun May 28 08:31:55 2023 Initialization Sequence Completed

```

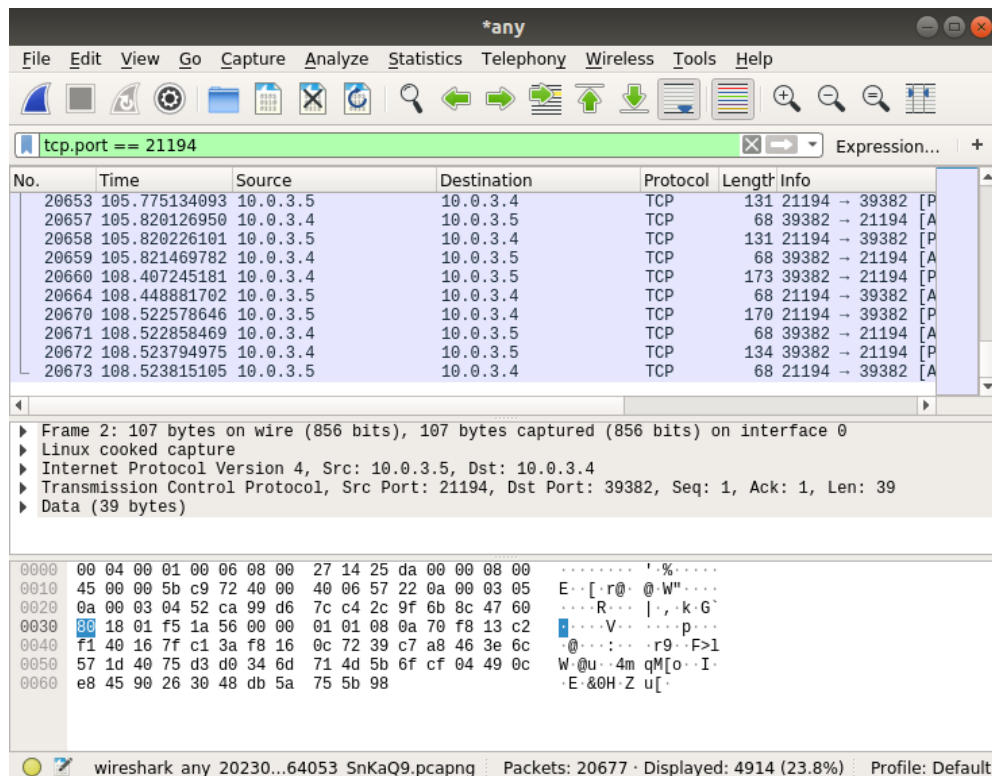
شکل ۱۴ - نحوه اتصال درست OpenVPN

می‌توان انتقال درست بسته‌ها را همانند شکل ۱۵ در Wireshark دید که در سمت کلاینت در حال اجراست. که همانطور که دیده می‌شود پورت 21194 را فیلتر کرده‌ایم و ترافیک بین دو VM با آدرس 10.0.3.4 و 10.0.3.5 که سرور و کاربر هستند را می‌بینید.



شکل ۱۵ - مشاهده بسته‌ها در سمت کلاینت

می‌توان بسته‌ها را در سمت server، در شکل ۱۶ مشاهده کرد.

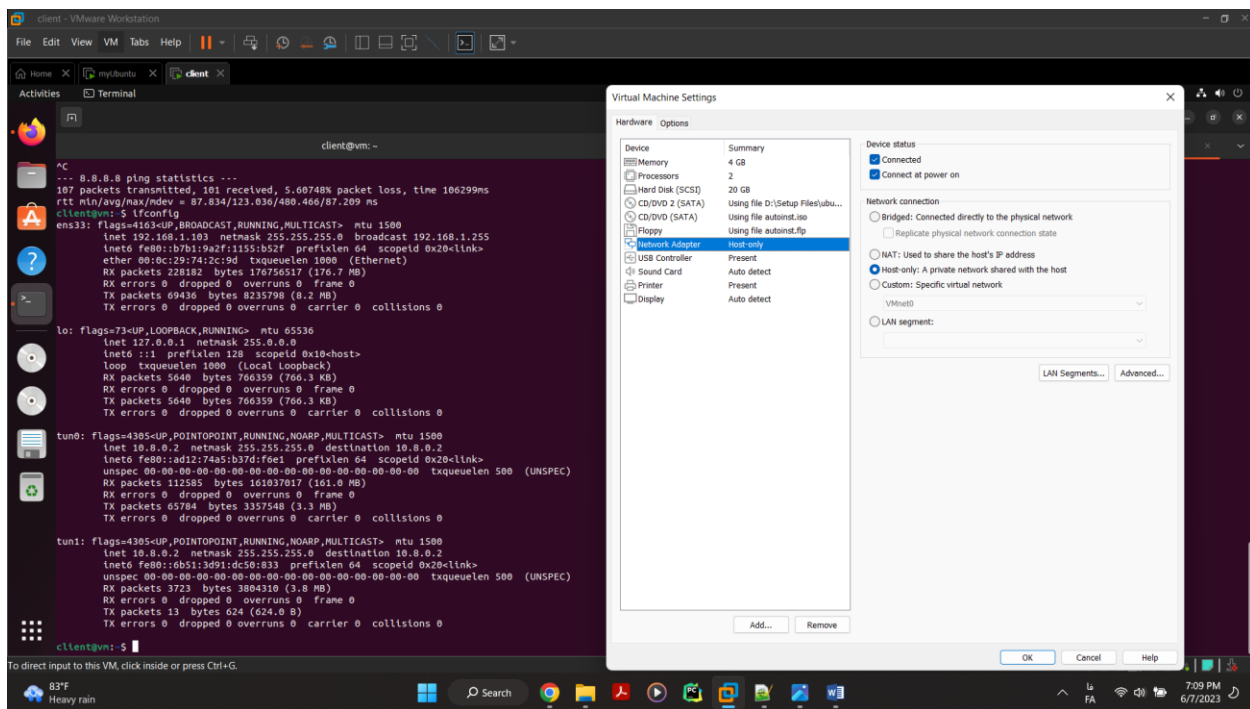


شکل ۱۶ - مشاهده بسته‌ها در سمت سرور

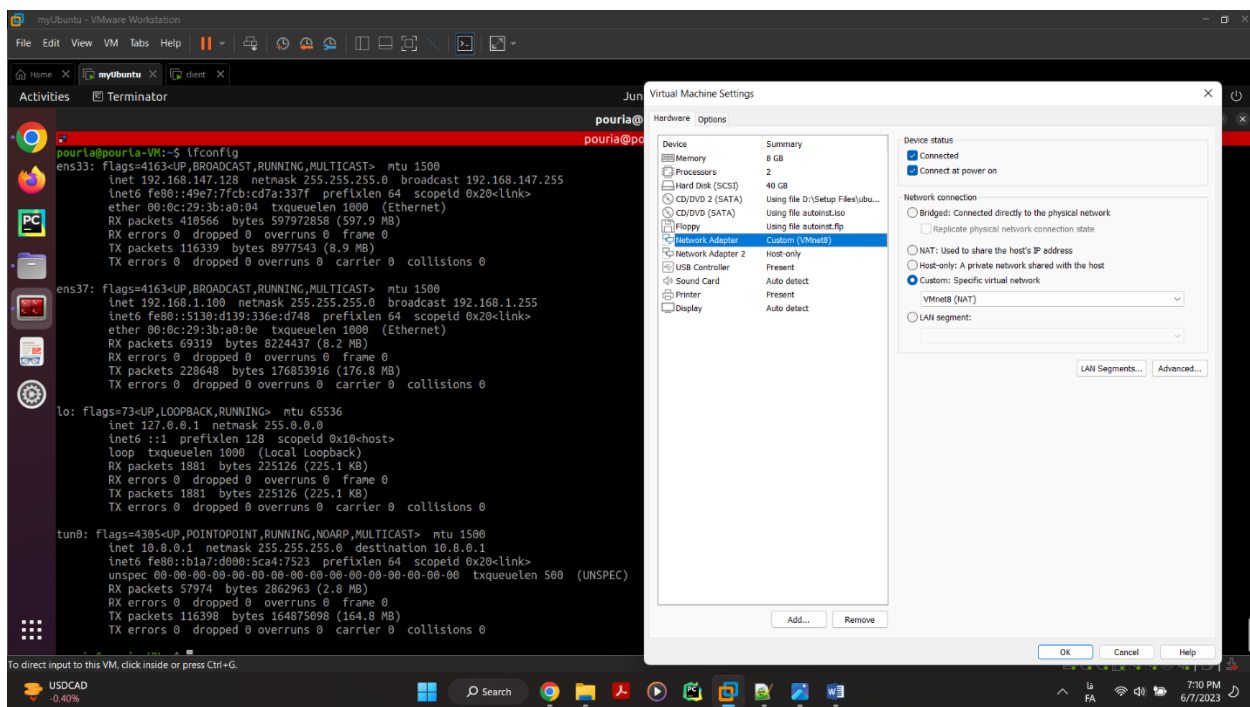
• روش دوم

برای پیاده‌سازی و تست VPN خود از دو ماشین مجازی Ubuntu استفاده کردیم که یکی نقش سرور و دومی کلاینت را دارد. برای ایجاد ارتباط از طریق VPN بین این دو از دو روش استفاده کردیم که هردو به درستی کار کردند:

کلاینت تنها دارای ارتباط Host-only با آیپی دستی تعریف شده دارد و سرور دارای یک اداپتور شبکه Nat برای ارتباط با اینترنت خارج دارد و یک اداپتور Host-only با آیپی دستی مشخص است. اکنون با ایجاد VPN و ست کردن کانفیگ آیپی دستی سرور و کلاینت، ترافیک موردنیاز کلاینت از بستر OpenVPN به سرور منتقل شده، از آنجا به بستر اینترنت متصل شده و پاسخ خود را نیز از همین تونل دریافت می‌کند. به این صورت مطمئن هستیم که VPN به درستی کار می‌کند چراکه کلاینت به خودی خود هیچ ارتباطی با اینترنت ندارد. که می‌توان سمت کلاینت را در شکل ۱۷ و سمت سرور را در شکل ۱۸ مشاهده کرد.



شکل ۱۷ - نشان دهنده تنظیمات در سمت کلاینت



شکل ۱۸ - نشان دهنده تنظیمات در سمت سرور

در ادامه مراحل نصب و کانفیگ OpenVPN بر سرور و کلاینت را توضیح می‌دهیم.

با اجرای دستورات زیر به ترتیب ابتدا پکیج OpenVPN را دریافت می‌کنیم سپس اجازه دسترسی و تغییر فایل نصبی را می‌دهیم و نهایتاً اجرای نصب و کانفیگ را استارت می‌زنیم. پس از اجرای دستور آخر مراحل اولیه کانفیگ را باید با توجه آیی، پورت و پروتکل‌های انتخابی خود انتخاب کنیم. نمونه ای از آپشن‌های نمایش داده شده را در شکل ۱۹ می‌بینیم.

```
$ wget https://git.io/vpn -O openvpn-install.sh
```

```
$ sudo chmod +x openvpn-install.sh
```

```
$ sudo bash openvpn-install.sh
```

```
Welcome to this OpenVPN road warrior installer!
```

```
Which protocol should OpenVPN use?
```

- 1) UDP (recommended)
- 2) TCP

```
Protocol [1]: 1
```

```
What port should OpenVPN listen to?
```

```
Port [1194]:
```

```
Select a DNS server for the clients:
```

- 1) Current system resolvers
- 2) Google
- 3) 1.1.1.1
- 4) OpenDNS
- 5) Quad9
- 6) AdGuard

```
DNS server [1]: 2
```

```
Enter a name for the first client:
```

```
Name [client]: iphone
```

```
OpenVPN installation is ready to begin.
```

```
Press any key to continue...
```

شکل ۱۹ – تنظیمات نصب OpenVPN

در مراحل اولیه کانفیگ این تنظیمات را اعمال می‌کنیم:

پروتکل را TCP انتخاب می‌کنیم که از Obfsproxy پشتیبانی کند، DNS سرور را روی سیستم فعلی قرار می‌دهیم که از تونل VPN برای مسیریابی ترافیک بسته استفاده کند، IP ارتباطی را در حالت host only باید برابر آییی دستی تعریف شده قرار دهیم تا کلاینت از آن طریق با سرور در ارتباط باشد و در حالت Nat همان آییی سیستم، نوع رمزنگاری را برای سادگی و صرفاً تمرکز بر روی پیاده‌سازی VPN، به صورت passwordless قرار می‌دهیم و در نهایت پورت ارتباطی VPN را روی پورت دلخواه ۴۴۳ قرار دادیم که DPI از طریق پورت پیش فرض OpenVPN که ۱۱۹۴ است، آن را تشخیص ندهد و مانند سایر بسته‌های معمول TCP روی پورت ۴۴۳ باشد ولی این پورت عملاً مجازی و بی‌اهمیت خواهد شد چرا که ارتباط بین کلاینت و سرور از طریق پورت پروکسی صورت خواهد گرفت و وظیفه پروکسی خواهد بود که بسته‌های دریافتی از پورت‌های خروجی خود را به پورت VPN منتقل کند.

پس از اعمال تنظیمات اولیه و نهایتاً انتخاب کلاینت مخاطب VPN و اختصاص یک اسم به آن، سرور VPN ساخته می‌شود و فایل کانفیگ سرور و کلاینت قابل مشاهده خواهند بود.

```
$ sudo more /etc/openvpn/server/server.conf
```

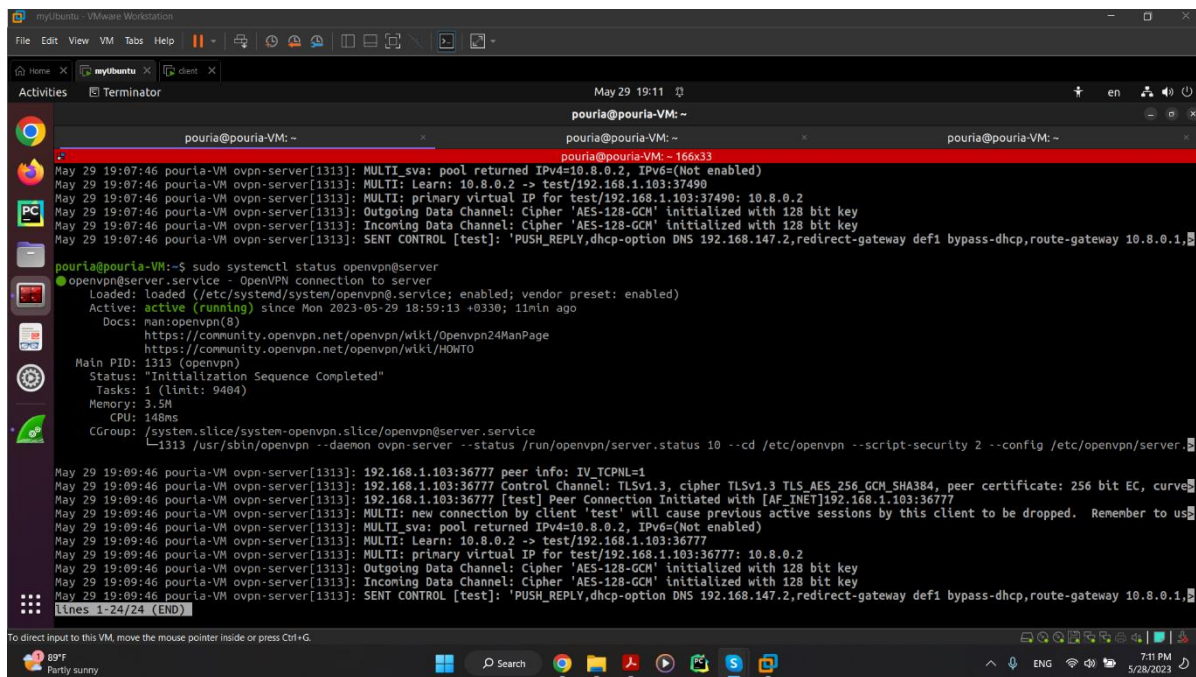
و فایل کانفیگ کلاینت هم در دایرکتوری home ساخته شده که باید محتویات آن را کپی کرده و در VM کلاینت در قسمت مربوطه قرار دهیم.

در آخر باید VPN سرور خود را start کنیم و status آن را مشاهده کنیم که به درستی بالا آمده باشد:

```
$ sudo systemctl start openvpn-server@server.service
```

```
$ sudo systemctl status openvpn-server@server.servic
```

اگر به درستی بالا آمده باشد خروجی مشابه لاگ در شکل ۲۰ خواهد داشت.



```
May 29 19:07:46 pouria-VM ovpn-server[1313]: MULTI_sva: pool returned IPv4=10.8.0.2, IPv6=(Not enabled)
May 29 19:07:46 pouria-VM ovpn-server[1313]: MULTI: Learn: 10.8.0.2 -> test/192.168.1.103:37490
May 29 19:07:46 pouria-VM ovpn-server[1313]: MULTI: primary virtual IP for test/192.168.1.103:37490: 10.8.0.2
May 29 19:07:46 pouria-VM ovpn-server[1313]: Outgoing Data Channel: Cipher 'AES-128-GCM' initialized with 128 bit key
May 29 19:07:46 pouria-VM ovpn-server[1313]: Incoming Data Channel: Cipher 'AES-128-GCM' initialized with 128 bit key
May 29 19:07:46 pouria-VM ovpn-server[1313]: SENT CONTROL [test]: 'PUSH_REPLY,dhcp-option DNS 192.168.147.2,redirect-gateway def1 bypass-dhcp,route-gateway 10.8.0.1,

pouria@pouria-VM:~$ sudo systemctl status openvpn@server
openvpn@server.service - OpenVPN connection to server
Loaded: loaded (/etc/systemd/system/openvpn@.service; enabled; vendor preset: enabled)
Active: active (running) since Mon 2023-05-29 18:59:13 +0330; 11min ago
Docs: man:openvpn(8)
https://community.openvpn.net/openvpn/wiki/Openvpn24ManPage
https://community.openvpn.net/openvpn/wiki/HOWTO
Main PID: 1313 (openvpn)
Status: "Initialization Sequence Completed"
Tasks: 1 (limit: 9404)
Memory: 3.5M
CPU: 148ms
CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
└─1313 /usr/sbin/openvpn --daemon ovpn-server --status /run/openvpn/server.status 10 --cd /etc/openvpn --script-security 2 --config /etc/openvpn/server.s
```

شکل ۲۰ - لاگ نصب OpenVPN

به این صورت راه اندازی سرور به اتمام رسیده است. حال به نصب VPN بر روی کلاینت می‌پردازیم. پس از نصب OpenVPN:

```
$ sudo apt install openvpn
```

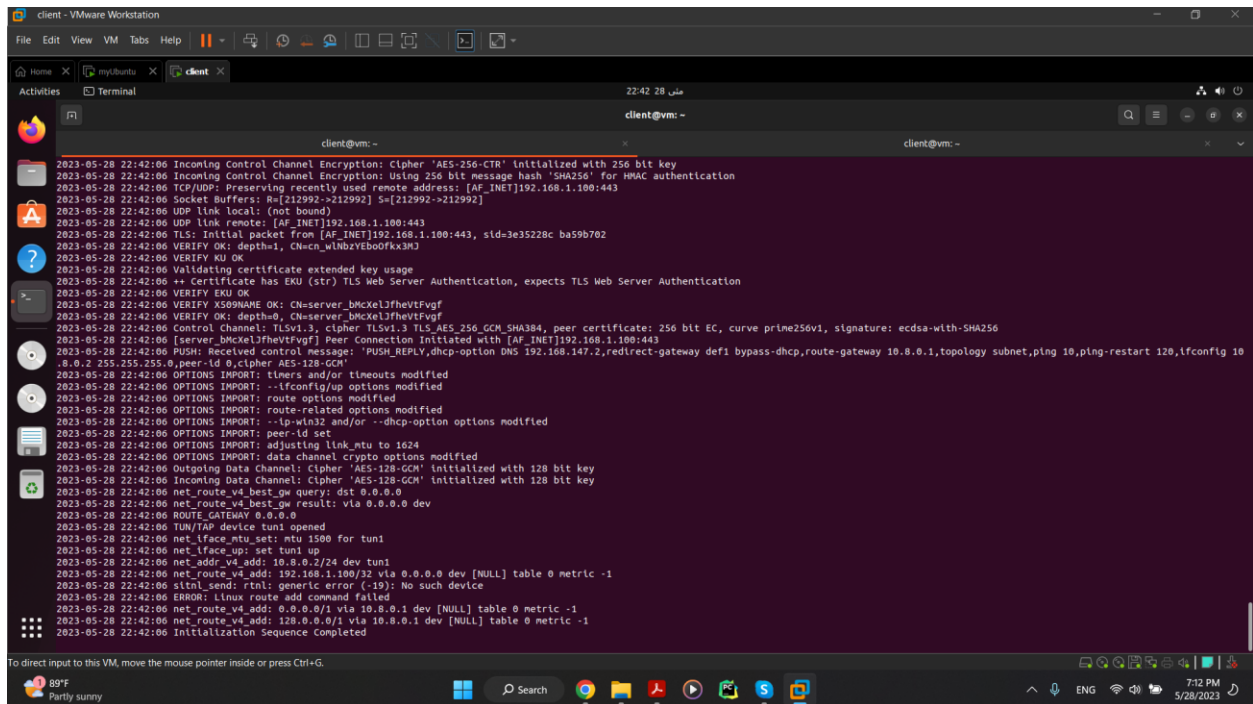
باید همانطور که گفته شد فایل کانفیگ کلاینت را در مسیر مربوط کپی کنیم:

```
$ sudo cp [client name].ovpn /etc/openvpn/client.conf
```

در آخر VPN را روی کلاینت نیز راه‌اندازی می‌کنیم تا به صورت خودکار به سرور متصل شود:

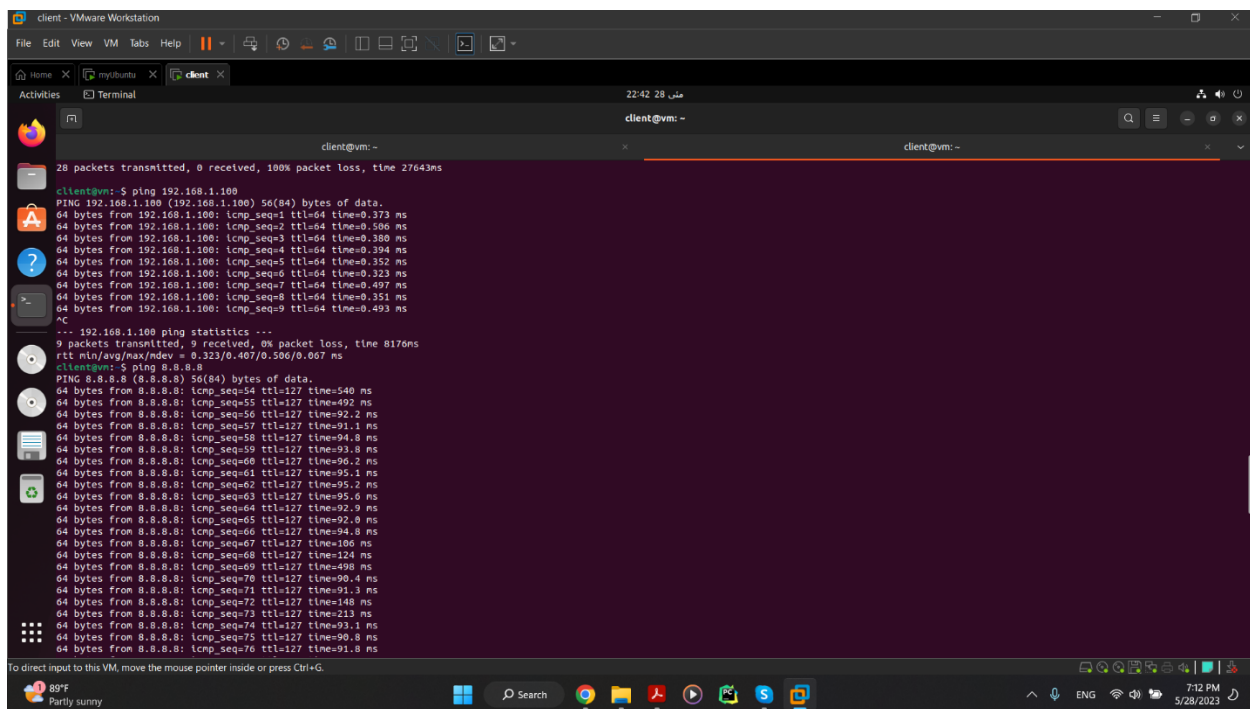
```
$ sudo openvpn --client --config /etc/openvpn/client.conf
```

و در صورت اتصال درست خروجی شکل ۲۱ را دریافت می‌کنیم.



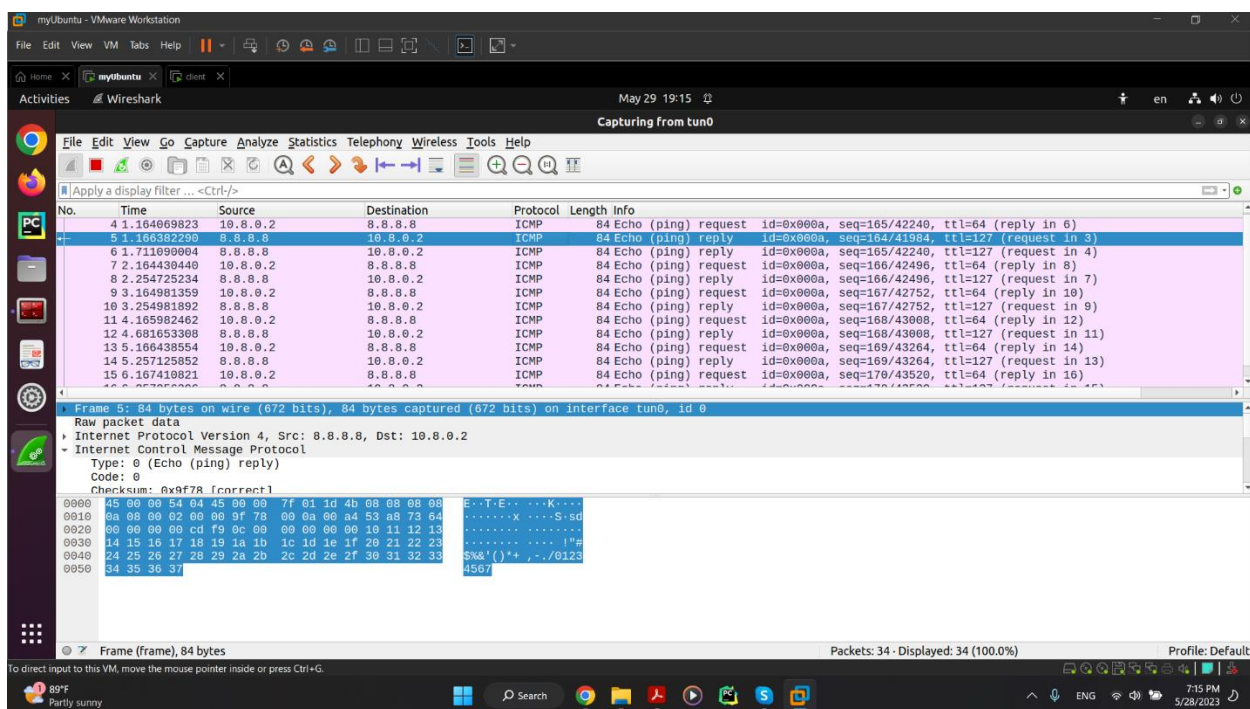
شکل ۲۱ – اتصال OpenVPN

برای تست درستی کارکرد اتصال VPN از سیستم host only کلاینت یک آیپی از اینترنت مانند 8.8.8.8 را پینگ می‌کنیم و خروجی مطلوب را دریافت می‌کنیم که در شکل ۲۲ قابل مشاهده است.



شکل ۲۲ - مشاهده اتصال به اینترنت از طریق شبکه VPN

و از طریق Wireshark چک می‌کنیم که بسته‌ها با مشخصات مدنظر (آییی مبدأ و مقصد تعریف شده تونل VPN در کانفیگ) انتقال می‌یابند که در شکل ۲۳ قابل مشاهده است.



شکل ۲۳ - نمایش بسته‌های در حال جابجایی در شبکه

نکته آخر که باید توجه کرد این است که ممکن است در حالت `host only` کلاینت، به درستی `DNS` سیستم کار نکند و برای مثال با اینکه `8.8.8.8` را پینگ می‌کند ولی `google.com` را دسترسی نداشته باشد که در این صورت باید دستی `nameserver` های سیستم را آپدیت کنیم و به طور مثال `nameserver` گوگل را به آن اضافه کنیم:

```
sudo nano /etc/resolv.conf
```

و خط زیر را اضافه کنیم و شبکه سیستم را ریست کنیم.

```
nameserver 8.8.8.8
```

```
sudo systemctl restart NetworkManager
```

اکنون که `VPN` به درستی کار می‌کند باید با اضافه کردن پروکسی همانطور که در روش اول اشاره شد آن را از دید `DPI` مخفی کنیم.

بخش چهارم: جمع‌بندی و بیان پیشنهادهای ادامه کار

در این پروژه به بیان مفهوم VPN پرداخته و سپس OpenVPN را به صورت تئوری بیان کردیم. در ادامه روش پیاده‌سازی آن را ذکر کردیم. به جهت اصلاح این پروتکل برای دور زدن فیلترینگ SPها، رویکردهایی مبنی بر handshaking و کپسوله‌سازی عنوان شد که روش استفاده از Obfsproxy تفصیل داده شد.

با توجه به راهکارهای ارائه شده و این که راهکار استفاده از Obfsproxy در پروژه TOR همچنان بر روی فیلترینگ کشورهای مانند چین و ایران جوابگوست می‌توان گفت که این روش را می‌توان با احتمال بالایی برای دور زدن فیلترینگ استفاده کرد. به همین دلیل اقدام به استفاده از این روش بر روی VPS خارجی شد که به دلیل امکان شناسایی DPI در اولین Handshake و از کار افتادن روش این ریسک پذیرفته نشد. برای این که بتوانیم این روش را آزمایش کنیم باید یک سیستم DPI را بر راهکار خود تست کنیم که ۲ راه را پیشنهاد می‌دهیم:

۱- استفاده از یک VM دیگر به عنوان DPI

۲- استفاده از یک داکر به عنوان سرور

بخش پنجم: منابع

1. <https://2019.www.torproject.org/docs/pluggable-transport>
2. <https://openvpn.net/vpn-server-resources>