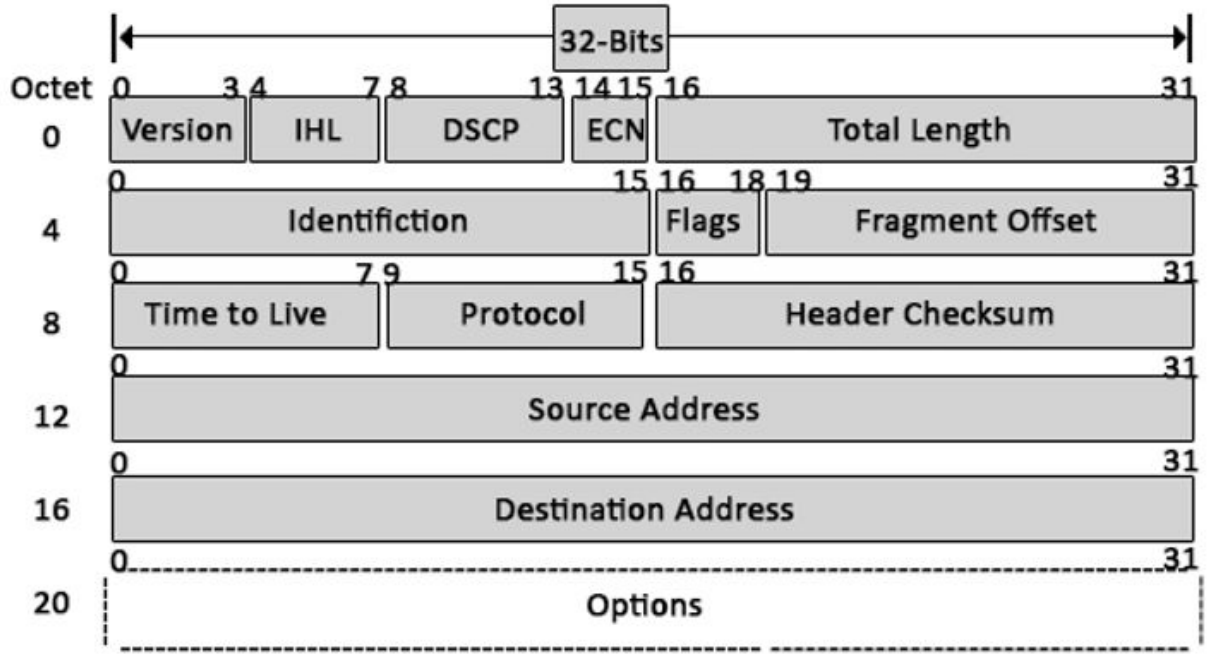
IPv4

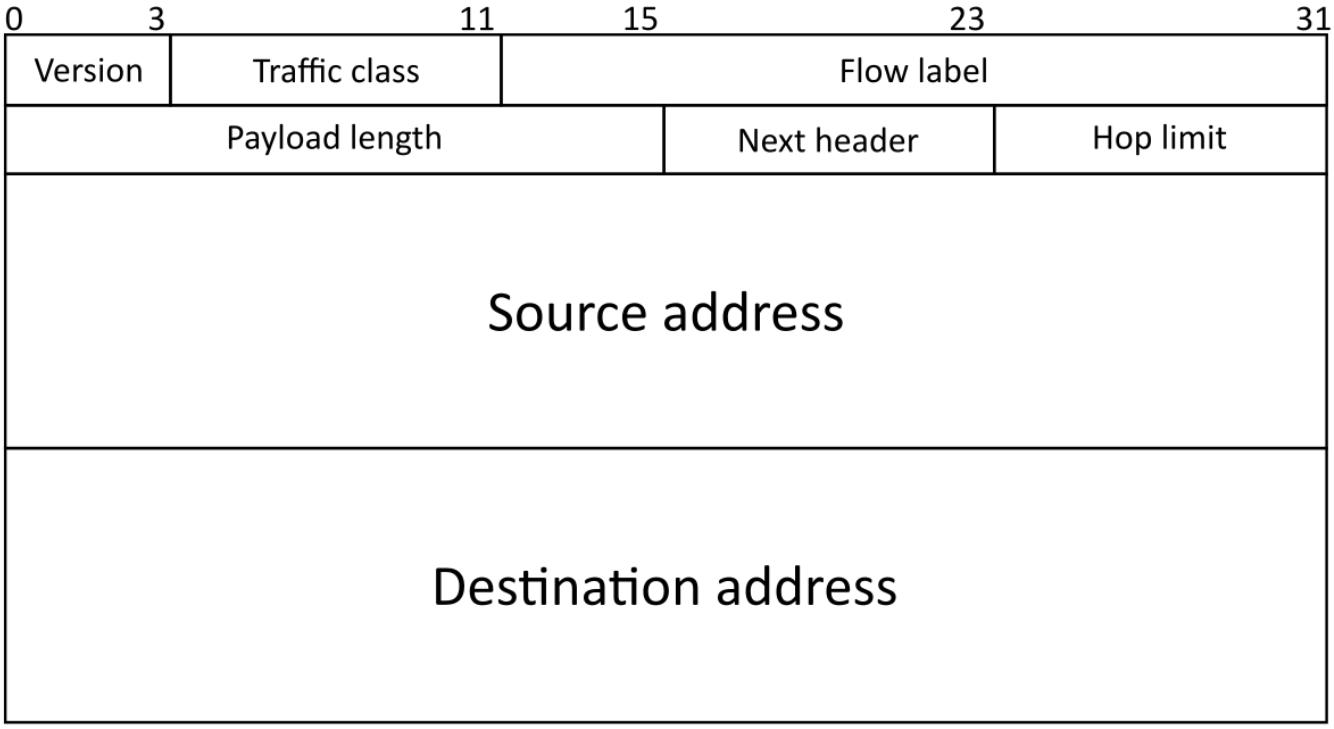


هدرهای بسته‌های نوع IPv4 از ۱۴ فیلد تشکیل شده‌اند که ۱۳ تای آن‌ها باید موجود باشند ولی آخری اختیاری می‌باشد:

1. اولین فیلد آن که ۴ بیت می‌باشد و ورژن نام دارد، فرمت هدر را مشخص می‌سازد که در این مورد همواره برابر ۴ است.
2. فیلد بعدی Internet Header Length می‌باشد (۴ بیت) که با توجه به متغیر بودن طول هدر و داشتن فیلدهای آپشنال، مشخص می‌سازد که طول هدر بسته ارسالی چقدر می‌باشد. مقدار آن برای این نوع بسته حداقل ۵ است (۵ *۳۲ بیت) و نهایتا می‌تواند ۳۲ باشد.
3. فیلد سوم که ۶ بیت است Differentiated Services Code Point نام دارد که با توجه به حجم و نوع ترافیک جابجایی معماری شبکه را مشخص می‌سازد تا برای در استفاده‌های متفاوت بتوان بهترین نتیجه را گرفت (با trade off میان سرعت و صحت و سلامت بسته‌ها).
4. فیلد بعدی Explicit Congestion Notification می‌باشد که از طریق آن روتر و سوییچ‌های میانی می‌توانند اندپوینت‌ها را از ترافیک زیاد باخبر کنند تا بتوانند بدون از دست‌دادن بسته ترافیک را مدیریت کنند.
5. فیلد پنجم طول کلی بسته را مشخص می‌سازد که شامل هدر و دیتای بسته می‌باشد. این طول به صورت مینیمم ۲۰ بایت و در بیشترین حد ۶۵۵۳۵ بایت می‌باشد.
6. فیلد ششم که Identification می‌باشد به منظور مشخص سازی گروه قطعه می‌باشد.
7. فیلد ۳ بیتی بعدی حاوی پرچم‌های کنترلی می‌باشد که بیت اول رزرو شده است و باید صفر باشد، بیت دوم برای مشخص کردن عدم قطعه قطعه کردن می‌باشد و بیت آخر مشخص می‌سازد که قطعه‌های بیشتری از آن بسته موجود می‌باشند.
8. فیلد بعدی offset قطعه را با توجه به ابتدای بسته قطعه قطعه نشده اصلی مشخص می‌سازد.
9. فیلد بعدی به منظور مشخص سازی تعداد پرش‌های معتبر می‌باشد و توسط نودهای میانی یک واحد یک واحد کاهش می‌یابد و در صورتی که قبل از رسیدن به مقصد برابر صفر شود، بسته دراپ می‌شود.
10. فیلد بعدی پروتکل مورد استفاده در لایه دیتا را مشخص می‌کند.
    a. ICMP 1, TCP 6 و UDP 17.

11. فیلد بعدی که checksum می‌باشد به منظور اطمینان از صحت هدر استفاده می‌شود.
12. فیلد بعدی آدرس مبدا می‌باشد.
13. فیلد آخر نیز آدرس مقصد است.
14. در نهایت می‌تواند فیلدهای آپشنال، با محدودیت ذکر شده قرار داد.(معمولا از این فیلدها استفاده‌ای نمی‌شود)

IPv6



در این ورژن هدر از ۸ فیلد تشکیل شده است:

1. فیلد اول به منظور مشخص کردن ورژن می‌باشد که در این مورد ۶ است.
2. فیلد دوم که ۸ بیت می‌باشد که ۶ بیت اول آن همان DSCP و ۲ بیت بعدی ECN هستند.(در بخش قبلی توضیح داده شدند)
3. فیلد بعدی توسط مبدا به منظور دسته بندی بسته‌ها، برای رسیدگی خاص و متفاوت به آن‌ها تعیین می‌شود.(برای مثال داده‌های Real time)
4. به منظور مشخص سازی طول دیتای ارسالی.
5. فیلد پنجم نوع هدر بعدی را مشخص می‌سازد که معمولا لایه انتقال آن می‌باشد.
6. تعداد پرش که همانند TTL در IPv4 می‌باشد.
7. دو فیلد آخر همانند ورژن ۴ آدرس مبدا و مقصد می‌باشد.

TCP

TCP Header				
Bits	0-15			16-31
0	Source port			Destination port
32	Sequence number			
64	Acknowledgment number			
96	Offset	Reserved	Flags	Window size
128	Checksum			Urgent pointer
160	Options			

1. مشخص کننده پورت مبدا می‌باشد.
2. مشخص کننده پورت مقصد می‌باشد.
3. فیلد سوم مشخص کننده شماره بسته به منظور دریافت بسته‌ها به صورت مرتب در مقصد می‌باشد.(و همچنین اطلاع از بسته‌های از دست رفته)
4. با توجه به سیاست کاری پروتکل، این فیلد از صحت دریافت بسته‌های ارسال شده و از طریق بیان کردن sequence number آن خبر می‌دهد.
5. سایز هدر این پروتکل را مشخص می‌سازد(در واقع محلی که دیتا در آن شروع می‌شود)
6. برای آینده رزرو شده است و به صورت پیش‌فرض صفر می‌باشد.
7. پرچم‌ها که شامل موارد زیر میباشد:

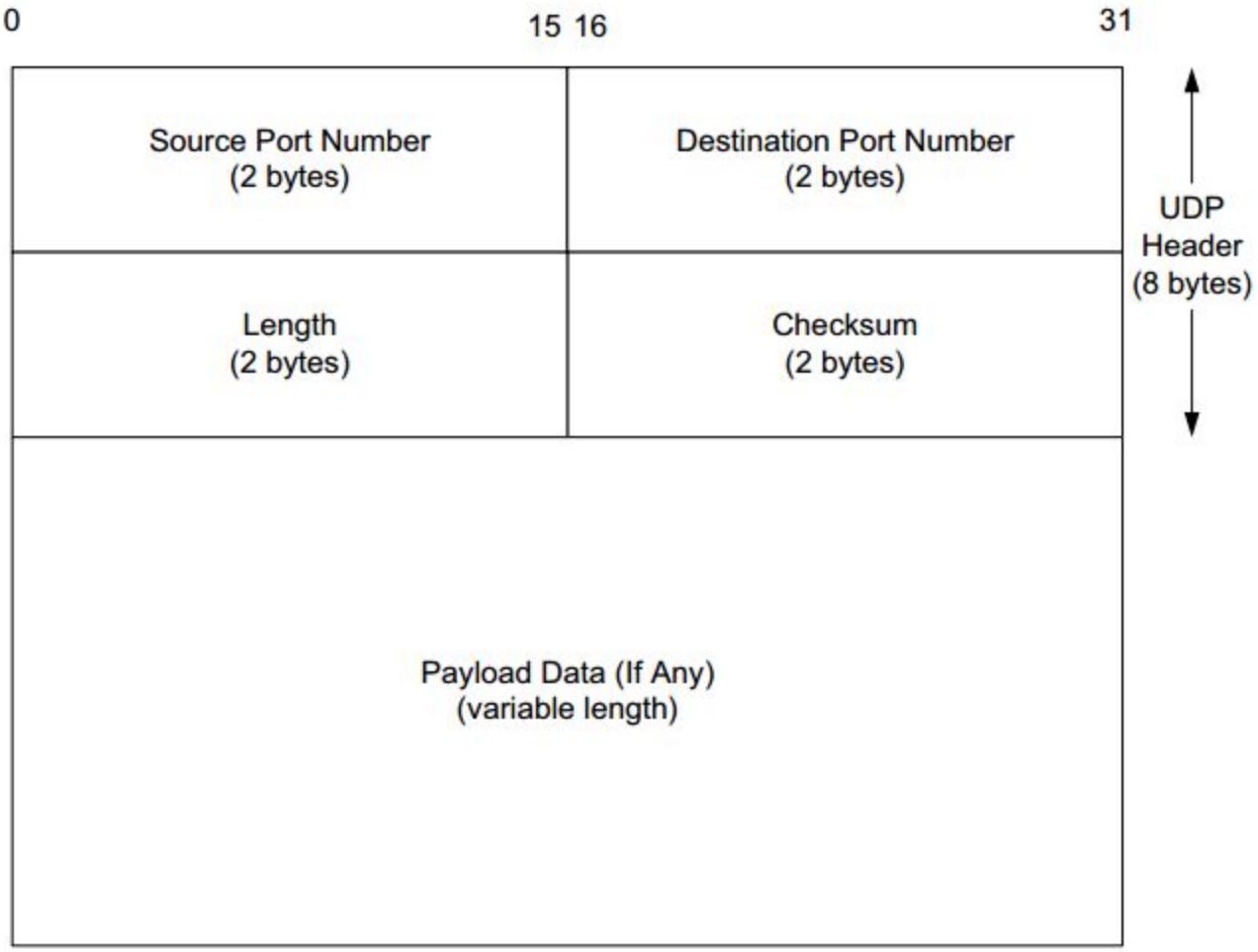
- NS (1 bit): ECN-nonce - concealment protection[a]
- CWR (1 bit): Congestion window reduced (CWR) flag is set by the sending host to indicate that it received a TCP segment with the ECE flag set and had responded in congestion control mechanism.[b]
- ECE (1 bit): ECN-Echo has a dual role, depending on the value of the SYN flag. It indicates:
    - If the SYN flag is set (1), that the TCP peer is ECN capable.
    - If the SYN flag is clear (0), that a packet with Congestion Experienced flag set (ECN=11) in the IP header was received during normal transmission.[b] This serves as an indication of network congestion (or impending congestion) to the TCP sender.
- URG (1 bit): Indicates that the Urgent pointer field is significant
- ACK (1 bit): Indicates that the Acknowledgment field is significant. All packets after the initial SYN packet sent by the client should have this flag set.
- PSH (1 bit): Push function. Asks to push the buffered data to the receiving application.
- RST (1 bit): Reset the connection
- SYN (1 bit): Synchronize sequence numbers. Only the first packet sent from each

end should have this flag set. Some other flags and fields change meaning based on this flag, and some are only valid when it is set, and others when it is clear.

- FIN (1 bit): Last packet from sender

8. سایز پنجره مشخص کننده ماکسیمم بسته‌ای می‌باشد که فرستنده این بسته میل به دریافت آن را دارد.
9. فیلد بعدی به منظور اطمینان از صحت هدر و دیتای بسته tcp می‌باشد.
10. اگر پرچم URG در بخش‌های قبلی یک باشد به این معنا می‌باشد که محتوای این فیلد مشخص کننده آخرین بایت دیتای فوری در چه offset ای از sequence number قرار دارد.
11. هدرهای آپشنال و انتخابی در این بخش قرار میگیرند.

UDP



1. فیلد اول مشخص کننده پورت مبدا می‌باشد.
2. فیلد دوم مشخص کننده پورت مقصد می‌باشد.
3. در این فیلد طول بسته یعنی هدر آن به همراه دیتا مشخص می‌شود.
4. به منظور چک کردن خطا و اطمینان از صحت بسته استفاده می‌شود.

Transport Layer Security

TLS و جد آن یعنی SSL که اکنون منسوخ شده‌است، پروتکل‌های کریپتوگرافی می‌باشند که به منظور امنیت ارتباطات بر بستر شبکه طراحی شده‌اند.
هدف اصلی این پروتکل تامین حریم خصوصی و رعایت امانت داده‌ها میان دو یا تعداد بیشتری سیستم در حال ارتباط می‌باشد.
این ارتباطات میان کلاینت و سرور باید حداقل یکی از ویژگی‌های زیر را داشته باشند:

1. به دلیل رمزگذاری داده به صورت کریپتوگرافی متقارن ارتباط خصوصی و امن می‌باشد.
2. شناسایی طرفین ارتباط از طریق کریپتوگرافی کلید عمومی تصدیق شود.
3. ارتباط قابل اعتماد است چرا که هر پیام منتقل شده شامل تصدیق صحت و امانت می‌باشد.

در واقع سه اصل این پروتکل Encryption, Authentication و Integrity می‌باشد.
این پروتکل از متدهای متفاوتی به منظور مبادله کلیدها، رمزنگاری داده‌ها و تصدیق صحت پیام‌ها می‌باشد که با تنظیمات درست و دقیق می‌توان حتی از رمزگشایی پیام‌های گذشته جلوگیری کرد.
با توجه به اسم آن مشخصا مربوط به لایه انتقال می‌باشد.

IPSec

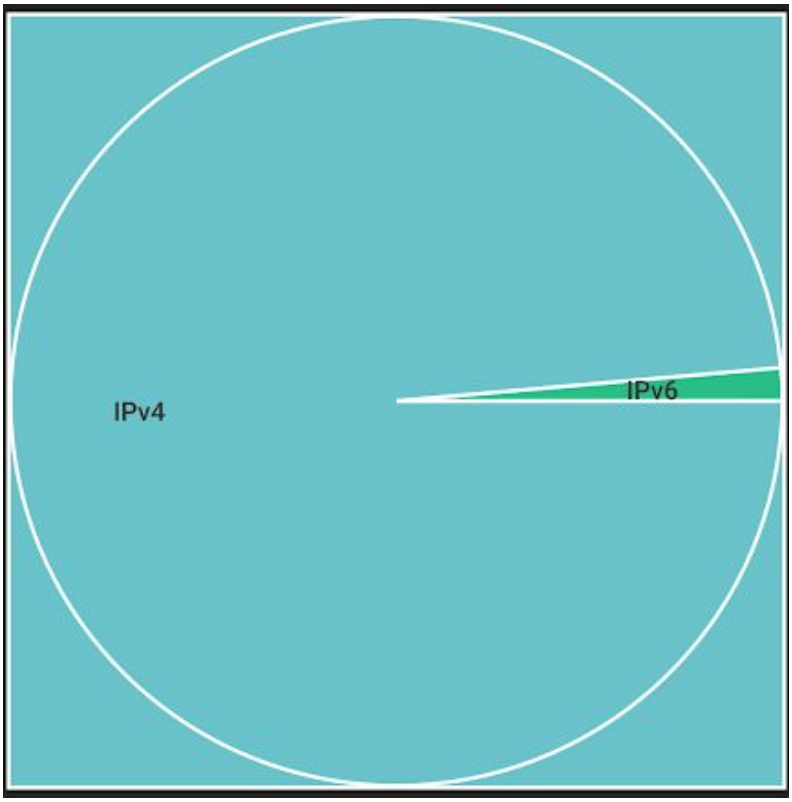
1. یک استاندارد تعریف شده توسط Internet Engineering Task Force می‌باشد که برای ارتباط میان دو endpoint تصدیق، امانت و صحت و محرمانگی داده را تامین می‌کند. همچنین بسته‌های رمزنگاری، رمزگشایی و تصدیق شده را تعریف می‌کند.
2. کاربردها:
    a. رمزگذاری داده‌های لایه اپلیکیشن
    b. تامین امنیت برای داده‌های مسیریابی ارسال شده توسط روترها بر روی اینترنت عمومی
    c. فراهم کردن تصدیق بدون رمزگذاری، مانند تصدیق داده‌ی ارسالی توسط فرستنده شناخته شده
    d. محافظت از داده‌های روی بستر شبکه با تنظیم کردن مدارهایی که در آن‌ها داده‌های ارسالی رمزگذاری شده‌اند گویی که ارتباط از طریق VPN می‌باشد
3. بزرگترین مزیت این استاندارد transparency آن برای لایه‌ی اپلیکیشن می‌باشد چرا که بر روی لایه‌ی اینترنت پیاده سازی شده است و روی لایه‌های بالایی خود تاثیری نمی‌گذارد. این استاندارد دارای امنیت بالایی می‌باشد.
4. از معایب آن می‌توان به این مورد اشاره کرد که زمانی که یک کامپیوتر بر این بستر وارد شبکه می‌شود، تمامی دستگاه‌های دیگر متصل به همان شبکه‌ی محلی نیز از طریق WAN می‌توانند به آن دسترسی داشته باشند و هر آسیب‌پذیری موجود در لایه IP می‌تواند در شبکه corporate توسط IPSec منتقل شود. از دیگر معایب آن می‌توان به این مورد اشاره کرد که اتصال به شبکه مورد نظر از خارج از محل همیشگی و به عبارتی off-site به دلیل تنظیمات firewall بسیار مشکل می‌باشد. همچنین این استاندارد بازده و performance سیستم را کاهش می‌دهد.

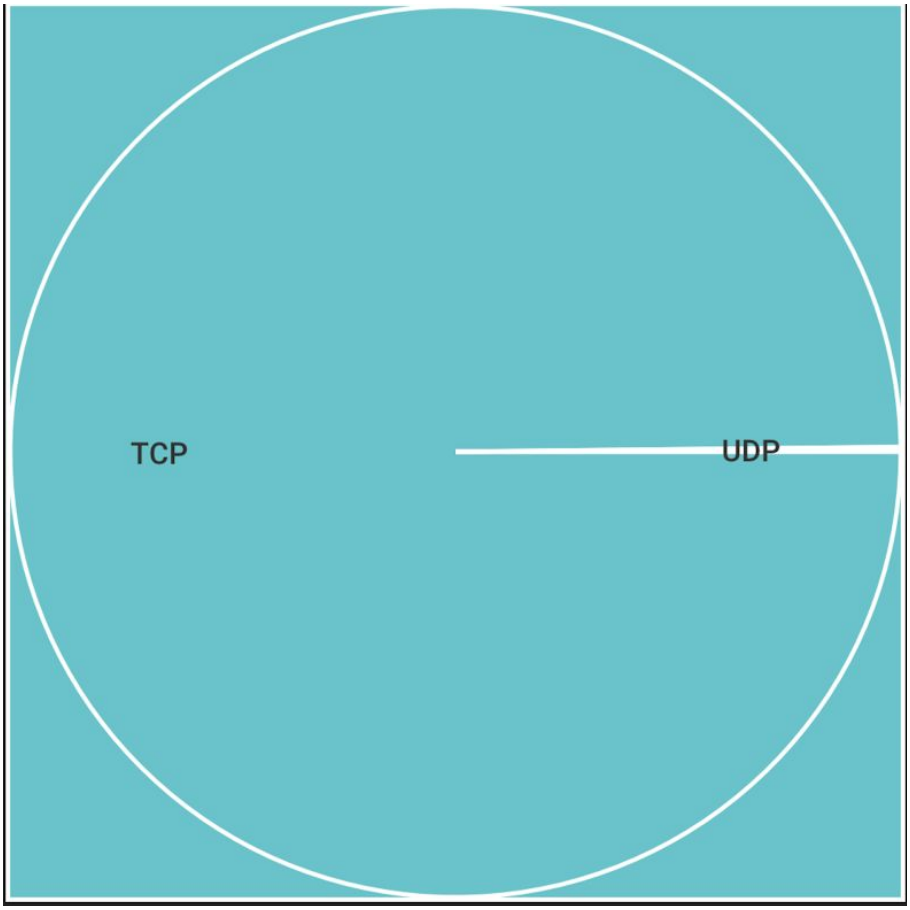
Packet Sniffer

```
1) Start Packet Sniffing
2) Set Target
3) Store Data
4) Exit
2
Please enter your target: 185.211.88.114
1) Start Packet Sniffing
2) Set Target
3) Store Data
4) Exit
3
Enter 1 to activate data storing and 0 to cancel it: 1
1) Start Packet Sniffing
2) Set Target
3) Store Data
4) Exit
1
1) \Device\NPF_{41254914-BBBE-4D22-9703-32656F98EE6C}: Microsoft
2) \Device\NPF_{9FF7539B-BD5E-40F9-99F5-709CE640A6F0}: Microsoft
3) \Device\NPF_{35222617-97E5-4E6C-9C40-64E3DDC17F30}: NdisWan Adapter
4) \Device\NPF_{062B9007-60BF-49AF-920B-8A8383AC6782}: Microsoft
5) \Device\NPF_{0D8E3DEB-4BDE-4708-A91F-79E025962131}: NdisWan Adapter
6) \Device\NPF_{5CB7DA63-A57A-4428-AA14-041A61021379}: NdisWan Adapter
7) \Device\NPF_Loopback: Adapter for loopback traffic capture
8) \Device\NPF_{A26A8ED0-DD18-4EA6-9798-DCC253BCFB58}: Realtek PCIe GbE Family Controller
Please enter device number or 0 to return to main menu: 2
Press enter to finish capturing
1) Start Packet Sniffing
2) Set Target
3) Store Data
4) Exit
Reports saved to: ./reports/20210104_140142
```

```
127.0.0.1 ==> 127.0.0.1: 5934
::1 ==> ::1: 80
192.168.1.101 ==> 239.255.255.250: 4
fe80::dd2a:ac6f:966e:166f ==> ff02::fb: 3
192.168.1.101 ==> 224.0.0.251: 3
```

```
Average length: 44
Min length: 44
Max length: 12260
---------------------------
Link Layer:

---------------------------
Network Layer:

IPv6: 83
IPv4: 5941
---------------------------
Transport Layer:

TCP: 6014
UDP: 10
---------------------------
Application Layer:

Payload: 2399
---------------------------
Fragments:

Dont Fragment: 5934
Fragment: 7
```






```
PACKET: 56 bytes, wire length 56 cap length 56 @ 2021-01-04 03:21:02.110221 +0330 +0330
- Layer 1 (04 bytes) = Loopback {Contents=[2, 0, 0, 0] Payload=[..52..] Family=IPv4}
- Layer 2 (20 bytes) = IPv4 {Contents=[..20..] Payload=[..32..] Version=4 IHL=5 TOS=0 Length=52 Id=20638 Flags=DF FragOffset=0 TTL=128 Protocol=TCP Checksum=0 SrcIP=127.0.0.1 D
- Layer 3 (32 bytes) = TCP  {Contents=[..32..] Payload=[] SrcPort=34717 DstPort=1080(socks) Seq=3510818502 Ack=0 DataOffset=8 FIN=false SYN=true RST=false PSH=false ACK=false U
PACKET: 56 bytes, wire length 56 cap length 56 @ 2021-01-04 03:21:02.417819 +0330 +0330
- Layer 1 (04 bytes) = Loopback {Contents=[2, 0, 0, 0] Payload=[..52..] Family=IPv4}
- Layer 2 (20 bytes) = IPv4 {Contents=[..20..] Payload=[..32..] Version=4 IHL=5 TOS=0 Length=52 Id=20639 Flags=DF FragOffset=0 TTL=128 Protocol=TCP Checksum=0 SrcIP=127.0.0.1 D
- Layer 3 (32 bytes) = TCP  {Contents=[..32..] Payload=[] SrcPort=34718 DstPort=1080(socks) Seq=1637574940 Ack=0 DataOffset=8 FIN=false SYN=true RST=false PSH=false ACK=false U
PACKET: 56 bytes, wire length 56 cap length 56 @ 2021-01-04 03:21:02.500257 +0330 +0330
- Layer 1 (04 bytes) = Loopback {Contents=[2, 0, 0, 0] Payload=[..52..] Family=IPv4}
- Layer 2 (20 bytes) = IPv4 {Contents=[..20..] Payload=[..32..] Version=4 IHL=5 TOS=0 Length=52 Id=20640 Flags=DF FragOffset=0 TTL=128 Protocol=TCP Checksum=0 SrcIP=127.0.0.1 D
- Layer 3 (32 bytes) = TCP  {Contents=[..32..] Payload=[] SrcPort=34719 DstPort=1080(socks) Seq=4061559979 Ack=0 DataOffset=8 FIN=false SYN=true RST=false PSH=false ACK=false U
PACKET: 56 bytes, wire length 56 cap length 56 @ 2021-01-04 03:21:02.513125 +0330 +0330
- Layer 1 (04 bytes) = Loopback {Contents=[2, 0, 0, 0] Payload=[..52..] Family=IPv4}
- Layer 2 (20 bytes) = IPv4 {Contents=[..20..] Payload=[..32..] Version=4 IHL=5 TOS=0 Length=52 Id=20641 Flags=DF FragOffset=0 TTL=128 Protocol=TCP Checksum=0 SrcIP=127.0.0.1 D
- Layer 3 (32 bytes) = TCP  {Contents=[..32..] Payload=[] SrcPort=34720 DstPort=1080(socks) Seq=3617287112 Ack=0 DataOffset=8 FIN=false SYN=true RST=false PSH=false ACK=false U
PACKET: 56 bytes, wire length 56 cap length 56 @ 2021-01-04 03:21:02.561642 +0330 +0330
- Layer 1 (04 bytes) = Loopback {Contents=[2, 0, 0, 0] Payload=[..52..] Family=IPv4}
- Layer 2 (20 bytes) = IPv4 {Contents=[..20..] Payload=[..32..] Version=4 IHL=5 TOS=0 Length=52 Id=20642 Flags=DF FragOffset=0 TTL=128 Protocol=TCP Checksum=0 SrcIP=127.0.0.1 D
- Layer 3 (32 bytes) = TCP  {Contents=[..32..] Payload=[] SrcPort=34721 DstPort=1080(socks) Seq=3149773963 Ack=0 DataOffset=8 FIN=false SYN=true RST=false PSH=false ACK=false U
PACKET: 56 bytes, wire length 56 cap length 56 @ 2021-01-04 03:21:02.565998 +0330 +0330
- Layer 1 (04 bytes) = Loopback {Contents=[2, 0, 0, 0] Payload=[..52..] Family=IPv4}
- Layer 2 (20 bytes) = IPv4 {Contents=[..20..] Payload=[..32..] Version=4 IHL=5 TOS=0 Length=52 Id=20643 Flags=DF FragOffset=0 TTL=128 Protocol=TCP Checksum=0 SrcIP=127.0.0.1 D
- Layer 3 (32 bytes) = TCP  {Contents=[..32..] Payload=[] SrcPort=34722 DstPort=1080(socks) Seq=3914915075 Ack=0 DataOffset=8 FIN=false SYN=true RST=false PSH=false ACK=false U
PACKET: 56 bytes, wire length 56 cap length 56 @ 2021-01-04 03:21:02.715945 +0330 +0330
- Layer 1 (04 bytes) = Loopback {Contents=[2, 0, 0, 0] Payload=[..52..] Family=IPv4}
- Layer 2 (20 bytes) = IPv4 {Contents=[..20..] Payload=[..32..] Version=4 IHL=5 TOS=0 Length=52 Id=20644 Flags=DF FragOffset=0 TTL=128 Protocol=TCP Checksum=0 SrcIP=127.0.0.1 D
- Layer 3 (32 bytes) = TCP  {Contents=[..32..] Payload=[] SrcPort=34723 DstPort=1080(socks) Seq=2242715705 Ack=0 DataOffset=8 FIN=false SYN=true RST=false PSH=false ACK=false U
PACKET: 56 bytes, wire length 56 cap length 56 @ 2021-01-04 03:21:03.059242 +0330 +0330
- Layer 1 (04 bytes) = Loopback {Contents=[2, 0, 0, 0] Payload=[..52..] Family=IPv4}
- Layer 2 (20 bytes) = IPv4 {Contents=[..20..] Payload=[..32..] Version=4 IHL=5 TOS=0 Length=52 Id=20645 Flags=DF FragOffset=0 TTL=128 Protocol=TCP Checksum=0 SrcIP=127.0.0.1 D
- Layer 3 (32 bytes) = TCP  {Contents=[..32..] Payload=[] SrcPort=34725 DstPort=1080(socks) Seq=1384954091 Ack=0 DataOffset=8 FIN=false SYN=true RST=false PSH=false ACK=false U
PACKET: 56 bytes, wire length 56 cap length 56 @ 2021-01-04 03:21:03.059301 +0330 +0330
- Layer 1 (04 bytes) = Loopback {Contents=[2, 0, 0, 0] Payload=[..52..] Family=IPv4}
```