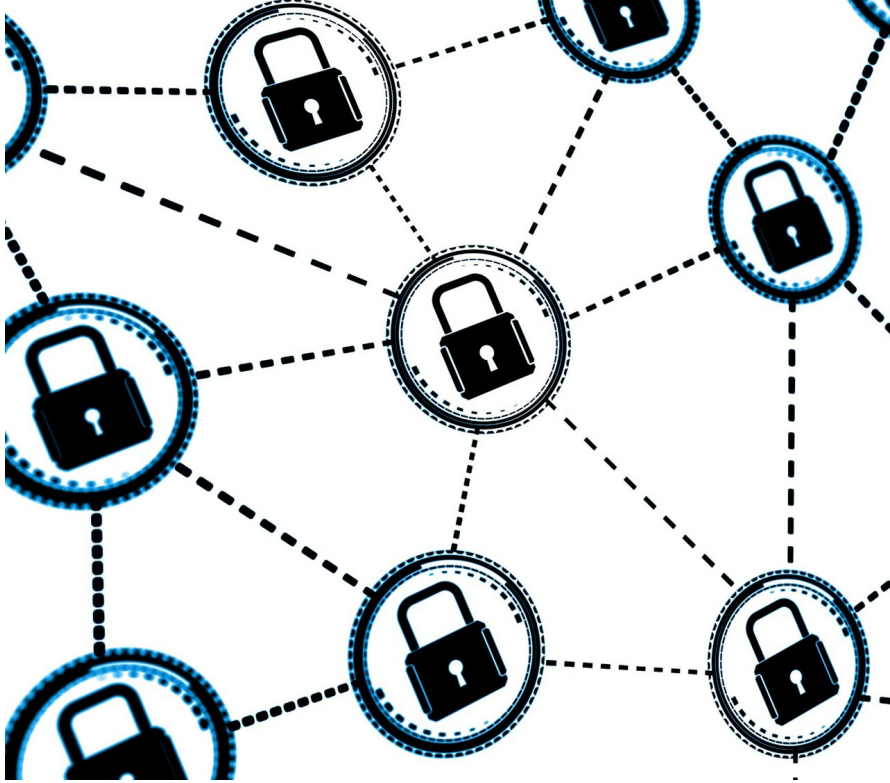
VPN or Vpwn? How Afraid Should You Be of VPN Traffic Identification?



An Empirical Analysis of VPN Traffic Fingerprinting in the Age of Censorship

Tanmay Rajore, Jithin S, Arnav Gupta, Keshav Gambhir, Anindya Prithvi, Dr. Sambuddho Chakravarty

IIIT Delhi

Network Traffic Measurement and Analysis (TMA 2025)

TMA CONFERENCE

INDRAPRASTHA INSTITUTE *of* INFORMATION TECHNOLOGY DELHI

VPN

Ideal Pathway

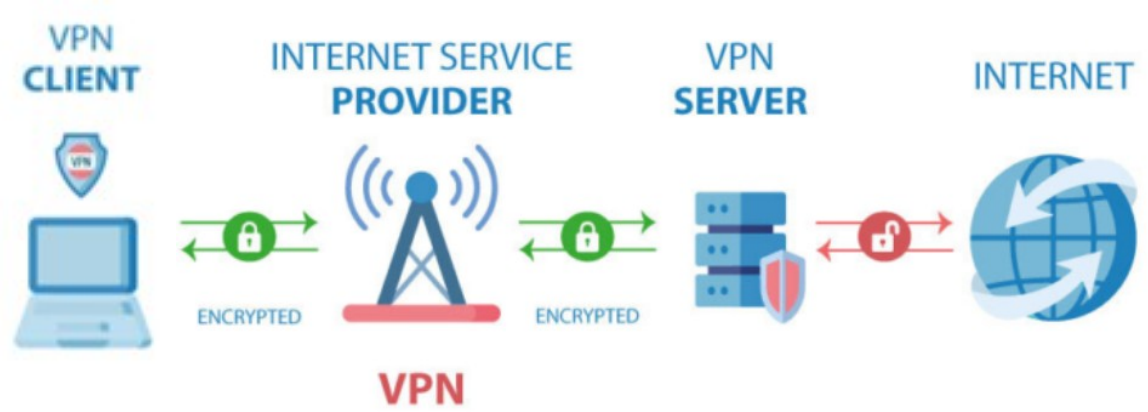


Image credit: sunteco

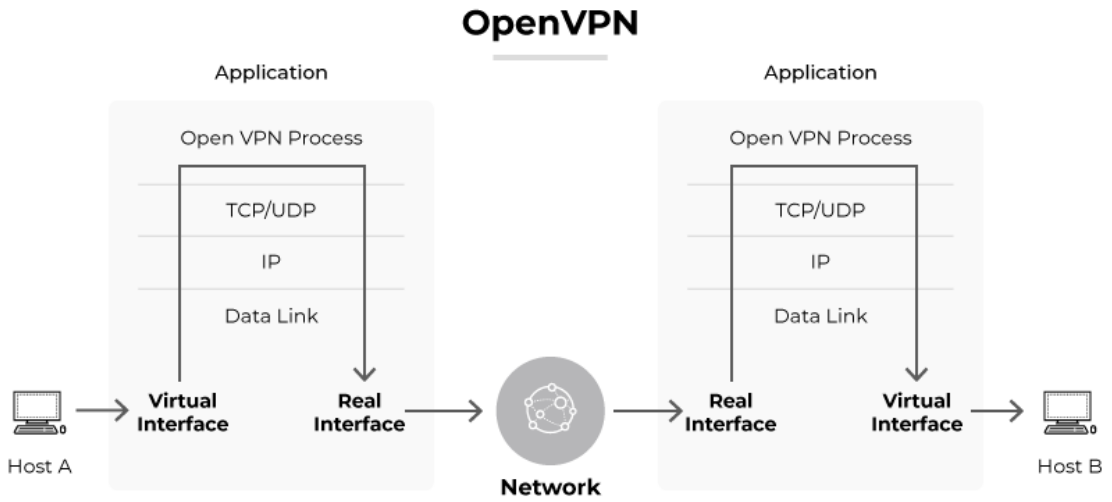
Protocols (in use today)

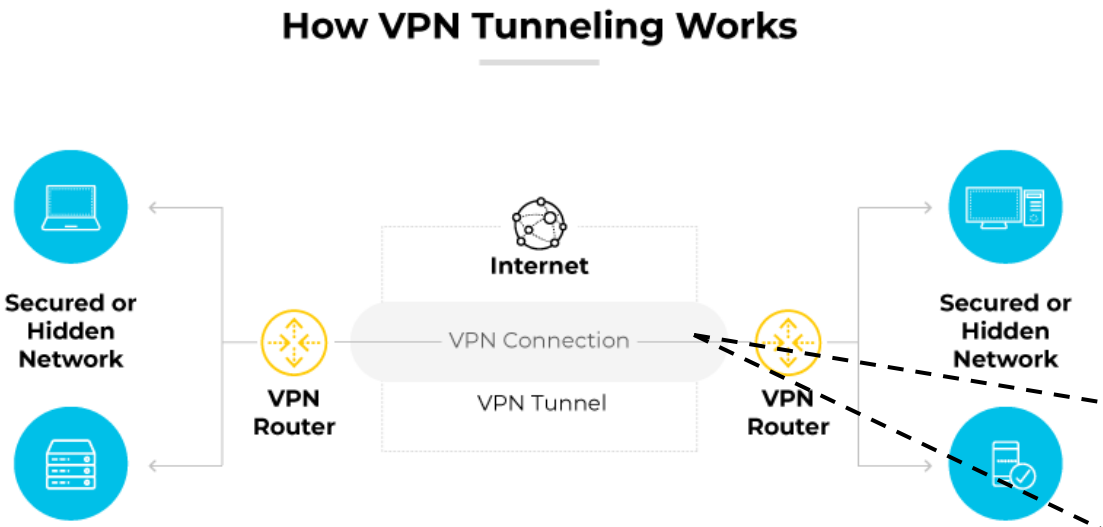
- OpenVPN
- Internet Key Exchange Version 2 (IKEv2)/IPSec
- WireGuard
- **TLS**

Older protocols (insecure/deprecated)

- Point-to-Point Tunneling Protocol (PPTP)
- Layer 2 Tunneling Protocol (L2TP)/IPSec
- Secure Socket Tunneling Protocol (SSTP)

VPN Tunneling





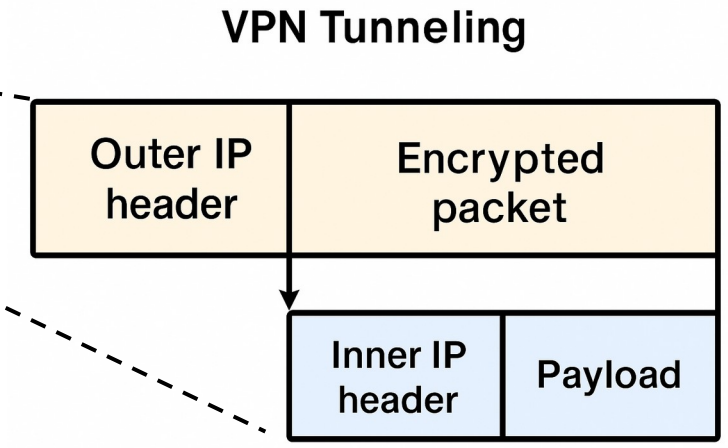


Image credit: palo-alto-networks[link]

VPN Identification

- **VPN Identification**
  - Process of detecting VPN traffic without decrypting it
  - Relies on identifying protocol patterns (headers, handshakes, traffic behavior
  - Used by ISPs and censors for blocking or surveillance
- **Xue et al. (Usenix'2022) –** "OpenVPN is open to fingerprinting"
  - OpenVPN: leaks control opcodes, easily fingerprinted

- **Frolov and Wustrow (2019) –** *"The Use of TLS in Censorship Circumvention"*
  - reveal SNI or cipher suite anomalies

Why This Matters

- Governments banning/monitoring VPNs
- Examples: China, Iran, India 2022, Russia
- Core question: Can VPN traffic be reliably identified using middleboxes and firewalls?

Links – [1][2][3]

Research Questions

- Can adversaries identify VPN traffic using DPI based middleboxes?
  - Based on current capabilities based on Datasheets and manual
  - Ex.- Fortinet OS/hardware offerings, Bluecoat Packetshaper series

- Which protocols are most/least vulnerable?
  - Most – easily identifiable (repeated patterns, plaintext identifiers)
  - Least – hard to detect using Firewalls and Middlesboxes current capabilities

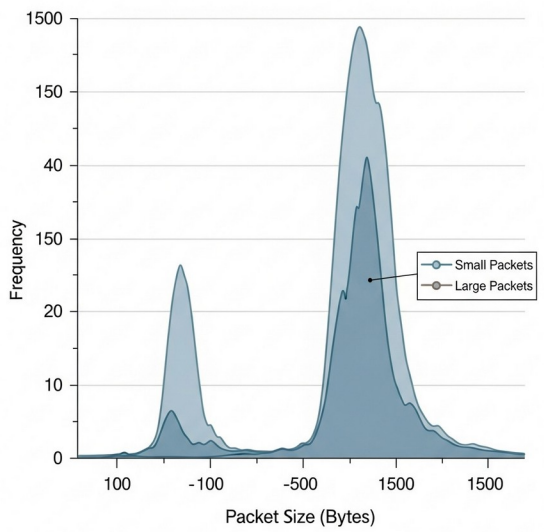
- What evasion strategies actually work?
  - Practical solutions and usability

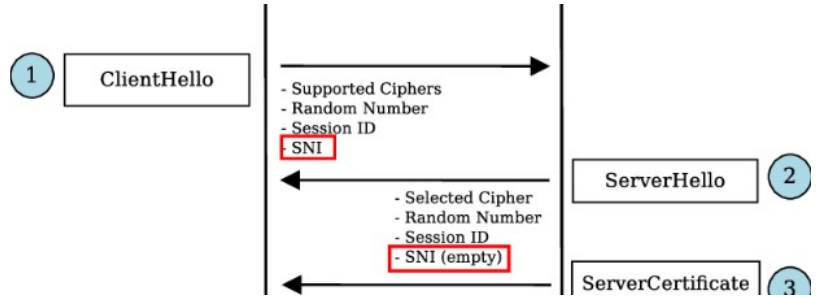
Threat Model

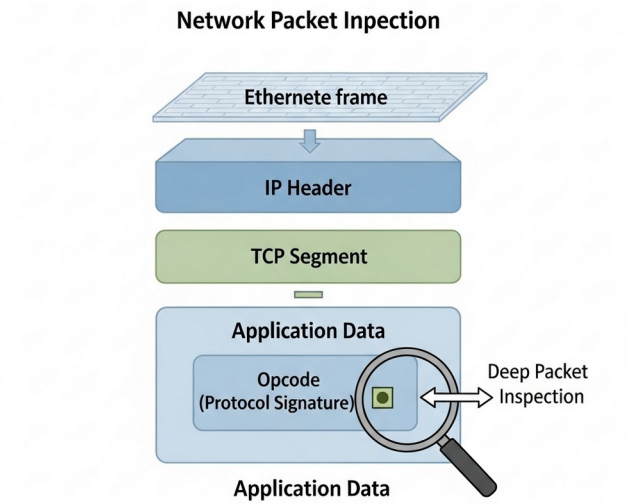
- ISP-level adversary firewalls/middleboxes
- DPI-enabled middleboxes – cannot parse at arbitrary offsets in real time.
- Can’t do large-scale ML on the middlebox or firewall at line rate.

Methodology Overview

- Choice VPNs
  - protocol diversity along with popularity
- Traffic Examination
  - Tools – MITMProxy, Wireshark
  - Kernel netfilter modules
- Markers we looked for?
  - SNI and plaintext identifiers
  - Packet size patterns
  - protocol signatures(opcodes)
  - Open ports and service providers
  - Application Reverse Engineering - flow of information
- Platforms: Windows, Android, Linux, MacOS







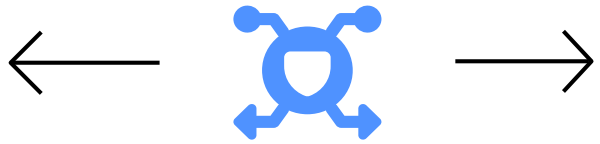
Logo rights belong to respective company , link1

Protocol Taxonomy

Open-Source VPNs

Freely available software with modifiable code and configurations.

Ex- OpenVPN, Wireguard



Proprietary VPNs

Vendor-specific extensions with custom features and undocumented protocols.

Ex- Lightway, Nordlynx, Chameleon

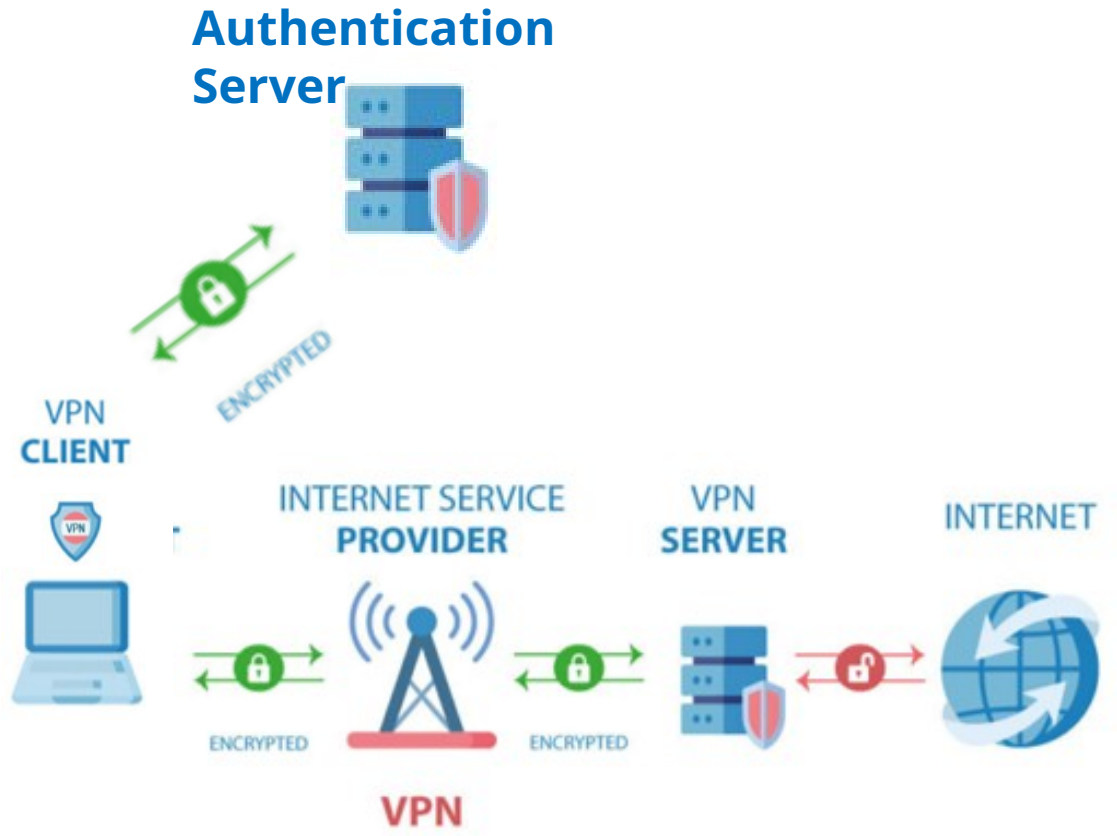
Commercial VPN connection

2 Step Process – client application (after login using username /password)

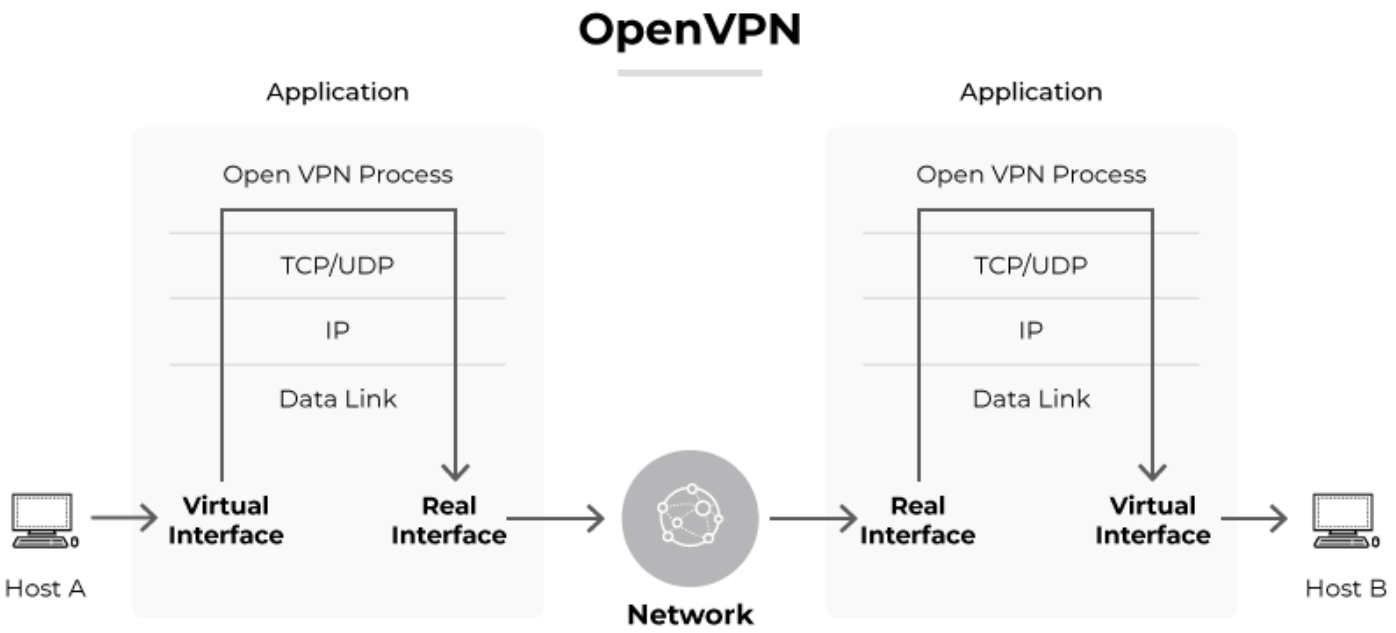
1. Authentication with Authentication server to fetch the VPN gateway details. (API calls)
    - Hosted by Major cloud provider

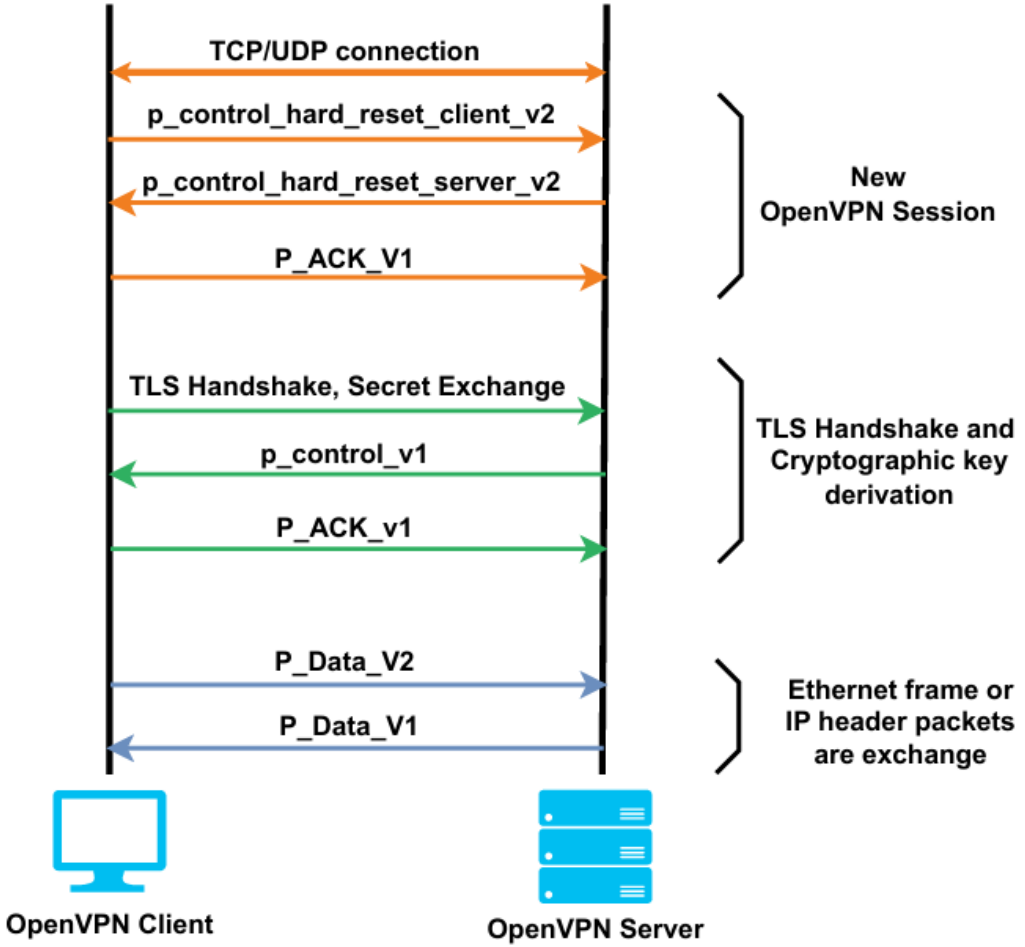
2. Connection to the VPN gateway
    - Using the data received from the auth server, the client application connects to specified gateways.

Weakness: An Adversary can just block the Authentication step



OpenVPN



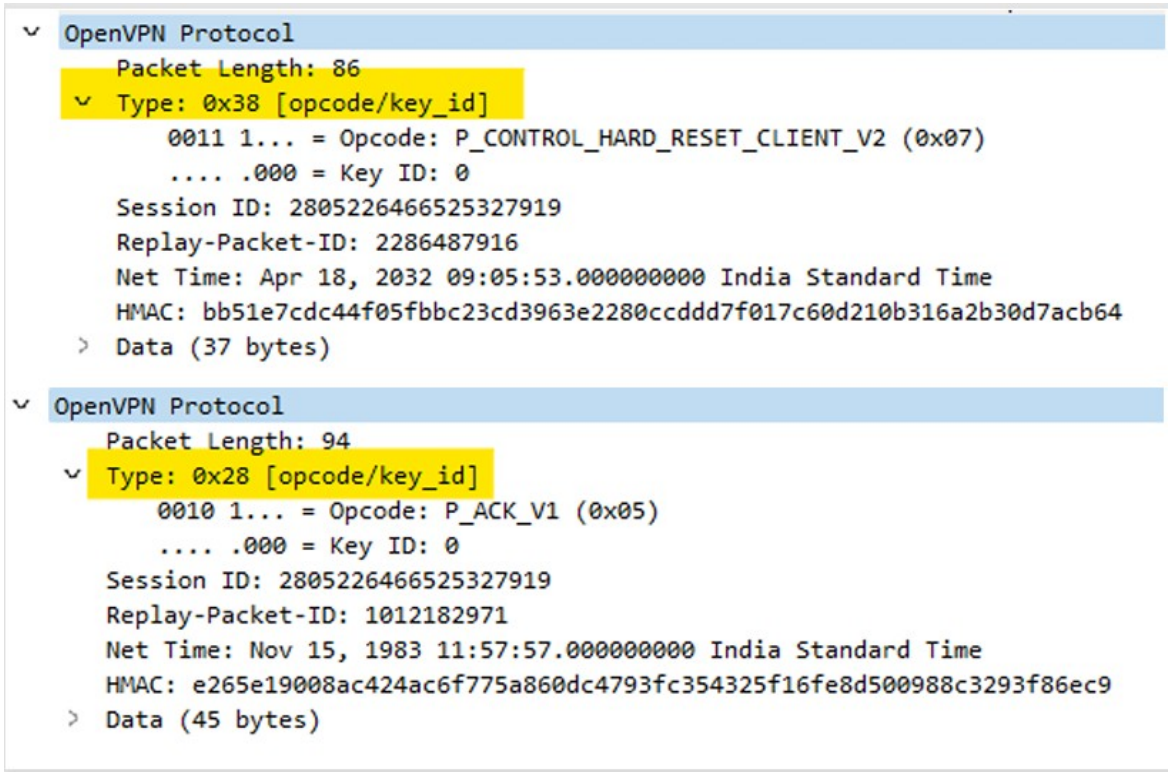


Vanilla – OpenVPN

- Encapsulation
- Handshake flow
- Opcodes

OpenVPN

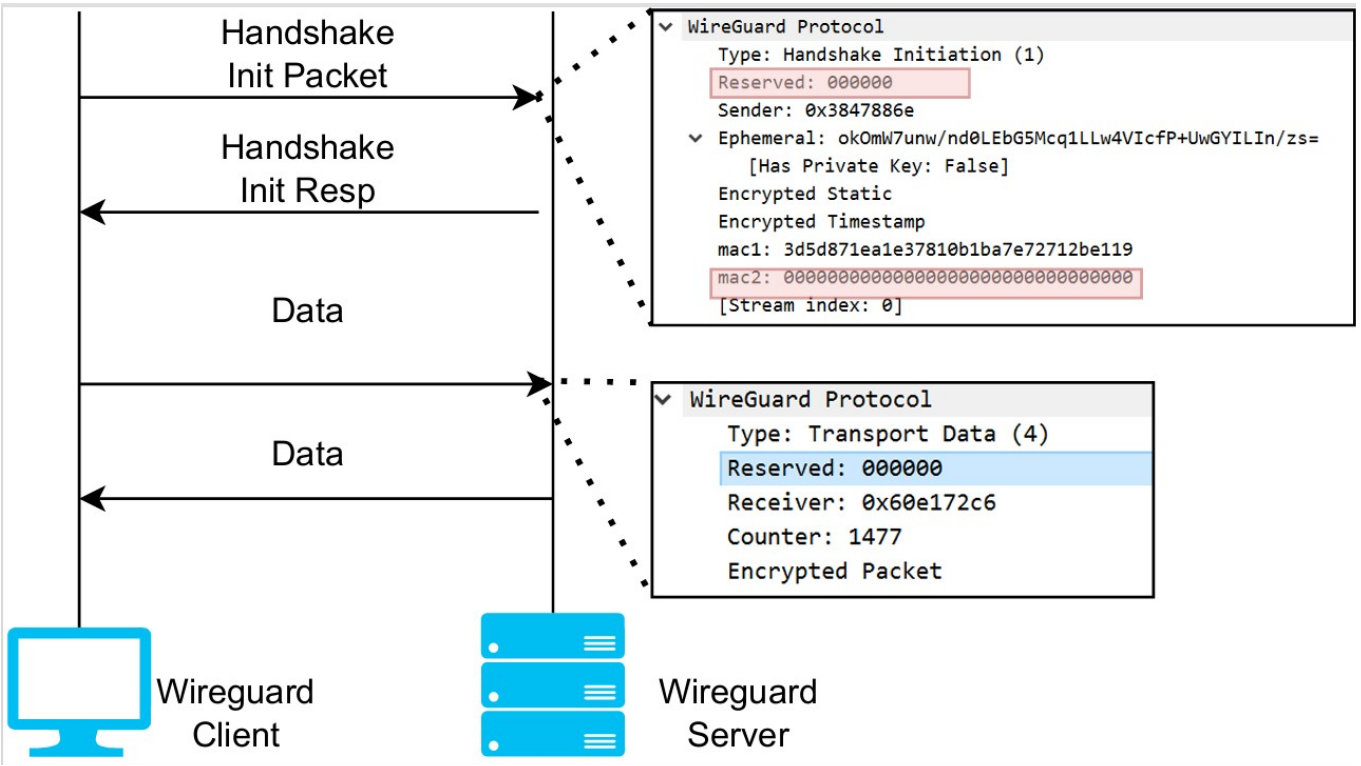
- Both the implementations of OpenVPN
  - Vanilla and commercial
    - opcodes including XOR patch (plaintext) [usenix'22]
    - XOR patch leaves 1 byte of opcode (XOR patch algorithm) [usenix'22] – harder for firewalls
- Malformed Packets – sending malformed HMAC packets get ignored by server
- Pre-shared Key mode
  - Complete packet is encrypted
  - Not Scalable
- New Protocol Proposal [towards the end]
  - Benefits of PSK + Scalable



Wireguard

- Next-Gen protocol, Lightweight, UDP based
- the user’s identity must be stored on the server and linked to an internal IP address assigned by the VPN

- **Weakness**
  - Plaintext identifiers
  - Fixed offset of repeated identifiers



Wireguard

- Commercial implementations provide only user level anonymization
- double NAT system allows NordLynx connection without storing any identifiable data on a server.
- Base Protocol – same **(No protection from Identification at protocol level)**

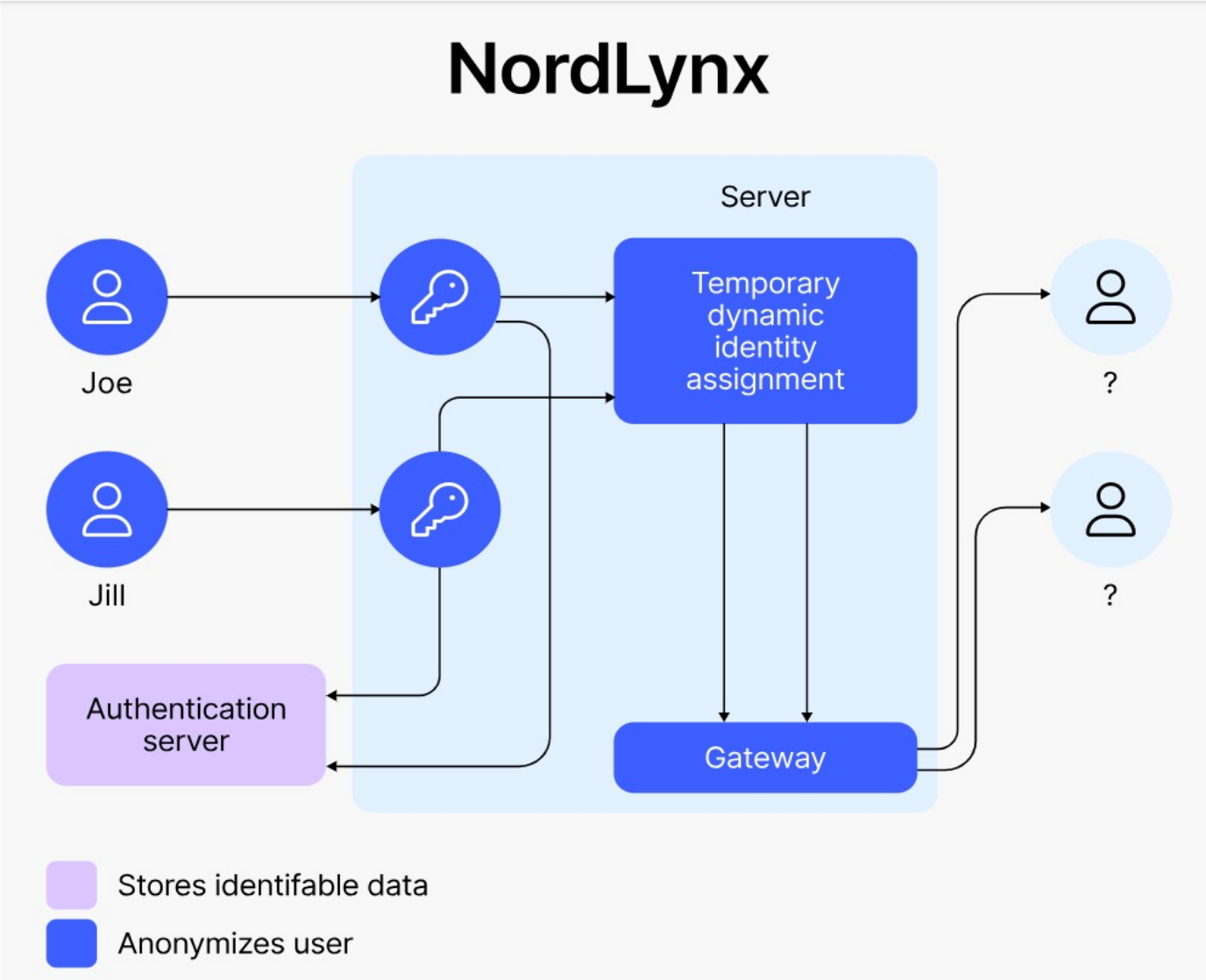


Image credits: [Link]

TLS and SSH-based VPNs

- **TLS 1.3**
  - Protocol for securing web and VPN traffic
  - Faster, more secure than TLS 1.2 (1-RTT handshake)
  - **Potential Markers:**
    - **SNI (Server Name Indication)**: still sent in plaintext (used for VPN detection)
    - **Number of Cipher Suites**
    - **Certificates encrypted**: prevents domain exposure post-handshake
  - Used in TLS-based VPNs (e.g., ProtonVPN Stealth, ExpressVPN Lightway)

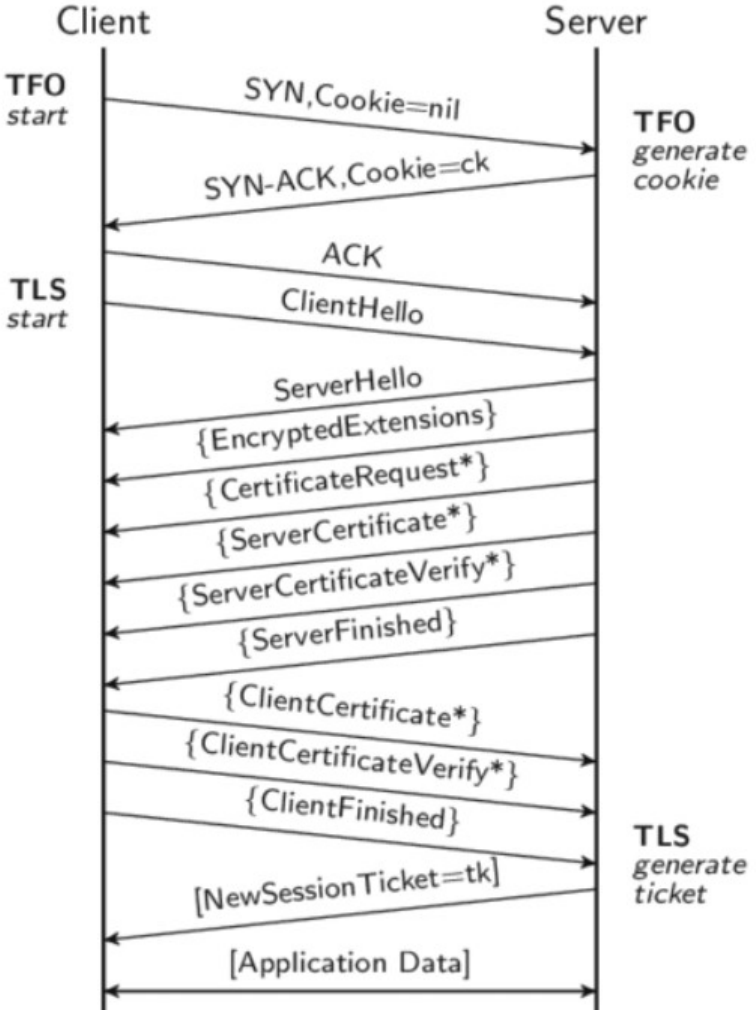


Image credits: [link]

TLS and SSH-based VPNs

- **SSH (Secure Shell)**
  - Secure channel for remote access and tunneling
  - Authenticates using keys or passwords
  - Encrypts all traffic, including handshakes
  - **Potential Markers:**
    - **Port 22** – default but can change
    - **SSH protocol version banner** (e.g., SSH-2.0-OpenSSH_8.2) during handshake
    - **Packet size & timing patterns**: regular keepalives, interactive keystrokes
  - Used in VPN obfuscation (e.g., OpenVPN over SSH)
  - Harder to distinguish from regular SSH traffic

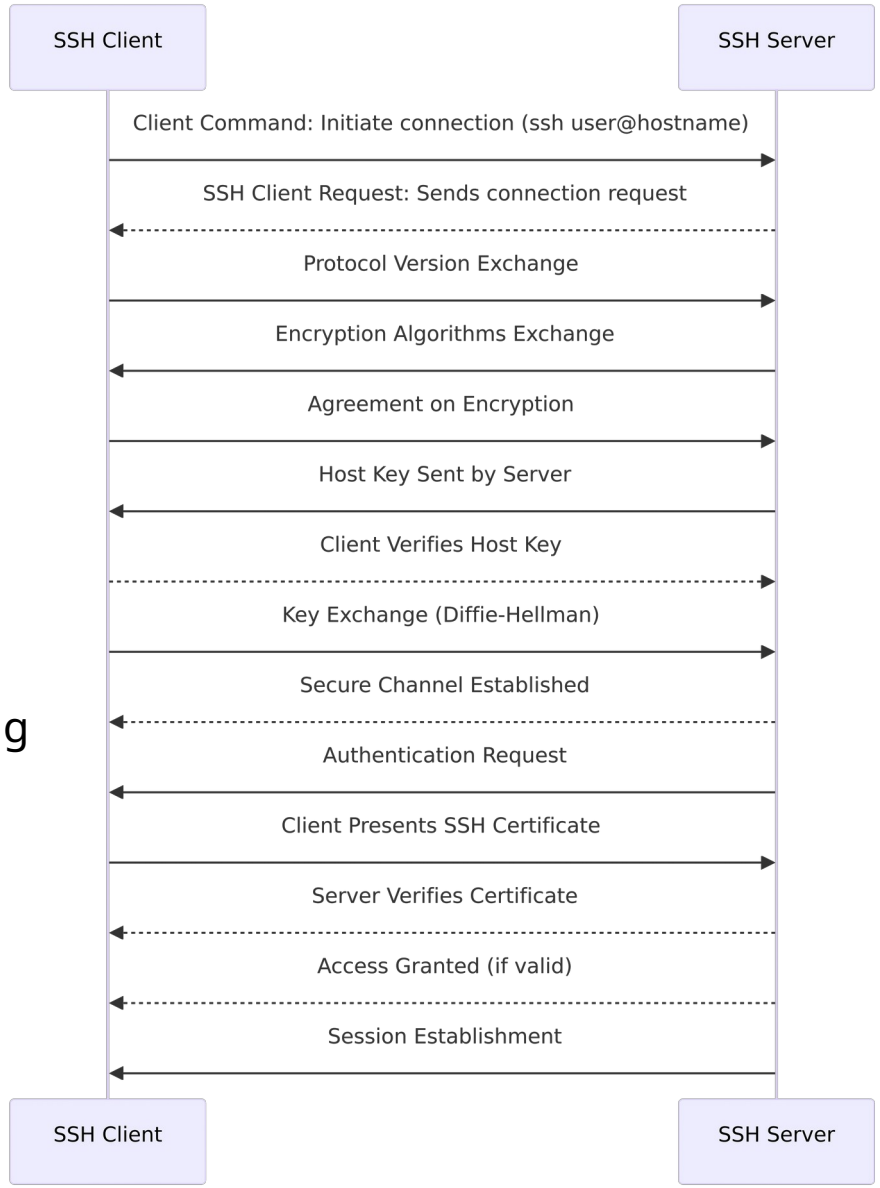
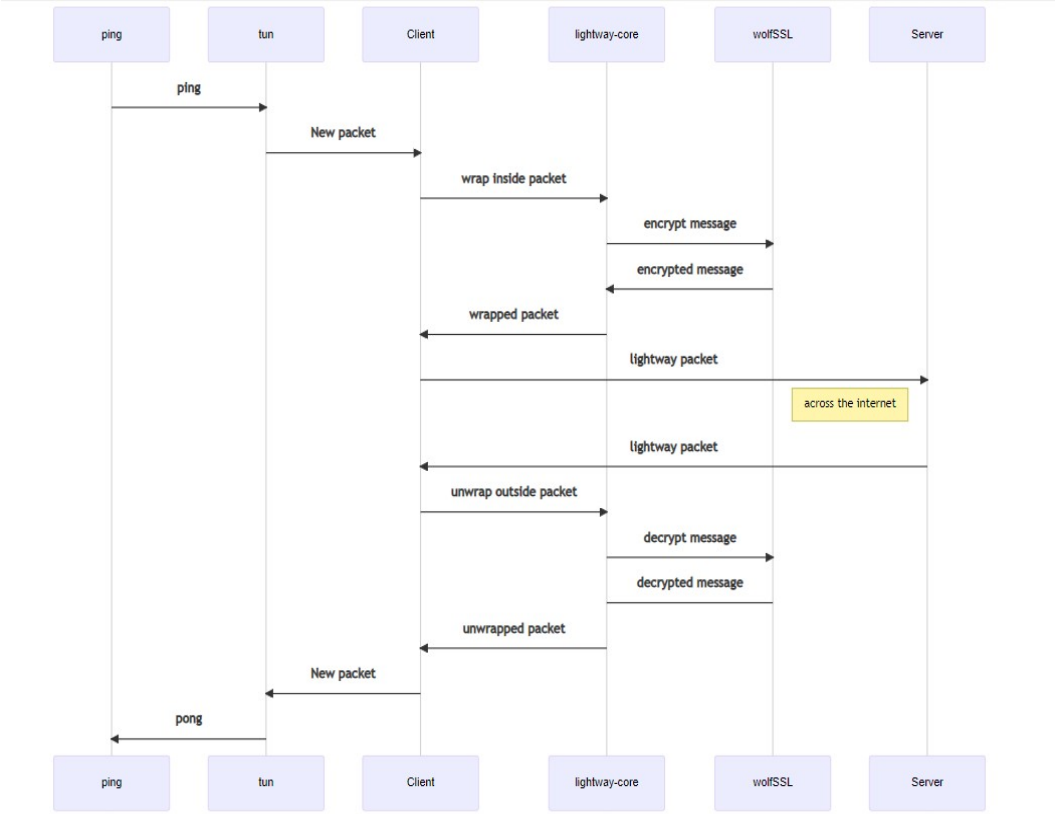


Image credits: [link]

TLS and SSH-based VPNs

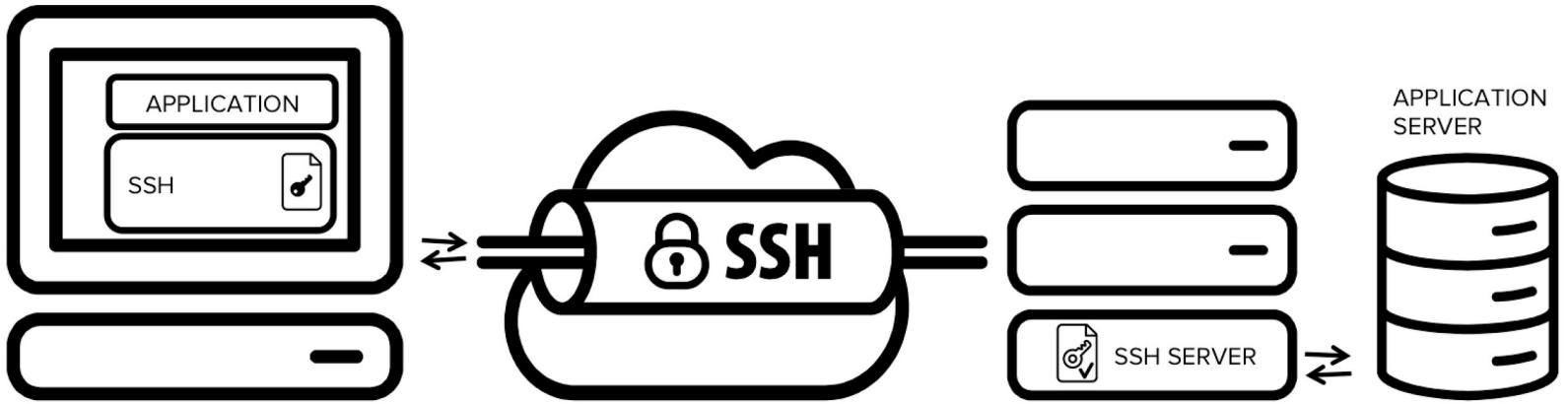
- **measures used by popular implementations**
  - Obfuscation via TLSv1.3
  - Randomized SNI fields (ex- ‘bdcyw.tr’,’ofd.es’)
  - No SNI (rare, detectable) [NDSS’19]
  - ProtonVPN Stealth, ExpressVPN Lightway (mimic Browser TLS)

    Browser TLS requests – not using only single cipher, legible SNI, ports



TLS and SSH-based VPNs

- SSH based observations
  - VyprVPN - Chameleon [SSH tunnelled OVPN]
  - Identical to SSH tunnel traffic [general SSH handshake markers]
  - Ciphers and handshake is SSH equivalent to standards
  - High collateral damage if blocked without analysis.



Evasion Techniques

- SNI poisoning
  - Change the SNI field to non-blocked Domains
    - Worked on Akami, fastly, Azure, Amazon hosted domains
    - Cloudflare – blocked – due to Cloudflare routing requirement for SNI matching with hostname check.

- MTU manipulation
  - Reducing the MTU from standard 1500 bytes to lower values in the range of 500.
  - Packet fragmentation
  - Evaded detection by Firewalls and Middleboxes without MITM
  - Minimal performance Hit for text based traffic.

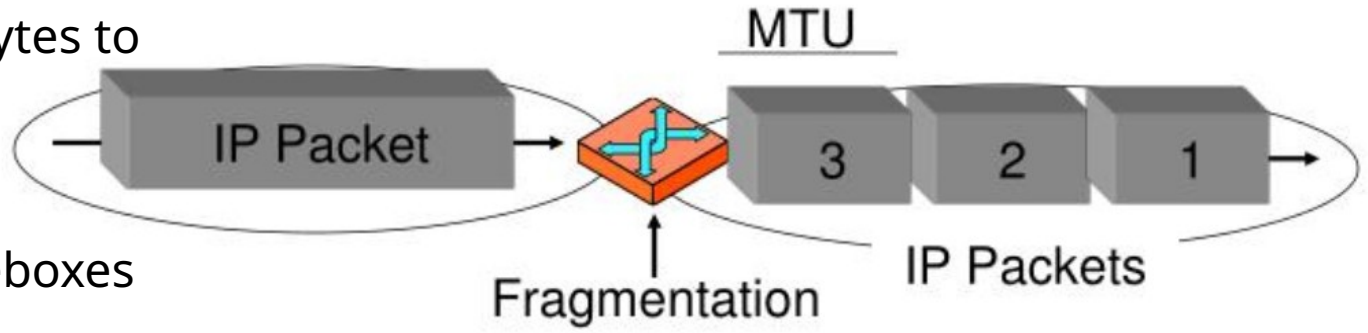
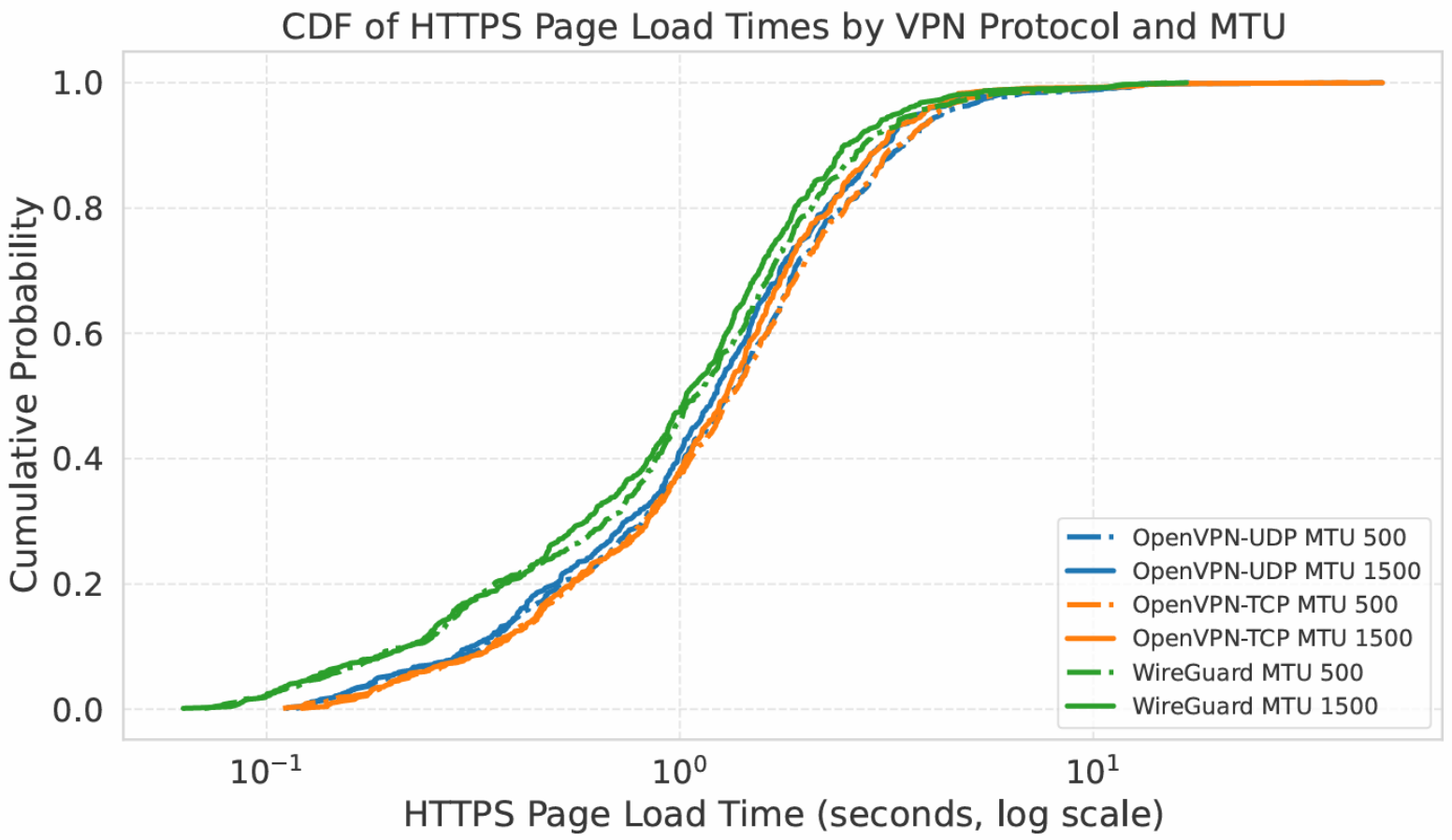


Image credits

Performance Results

- CDF graphs for MTU 500 vs 1500
- Tranco List top 1K
- Across open-source Protocols
  - OpenVPN TCP
  - OpenVPN UDP
  - Wireguard



Firewalls

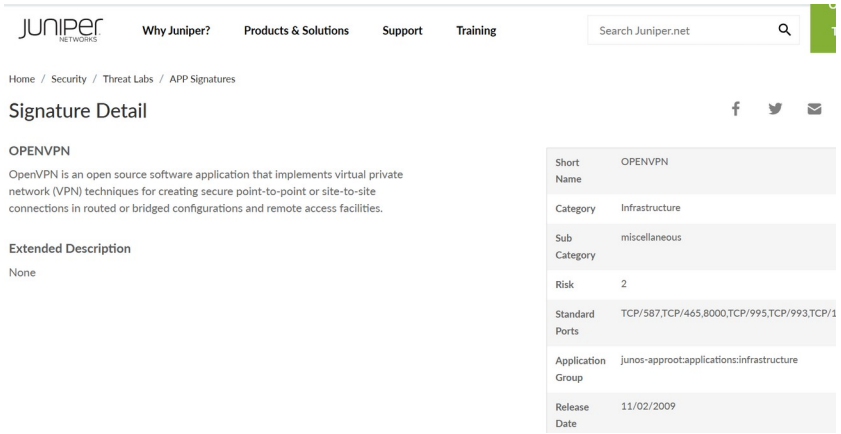
- Modern Firewalls - Static signatures
  [Fortinet, checkpoint, Bluecoat]
- Limited Configurability – specific fields
- Software based ML- solutions coming up

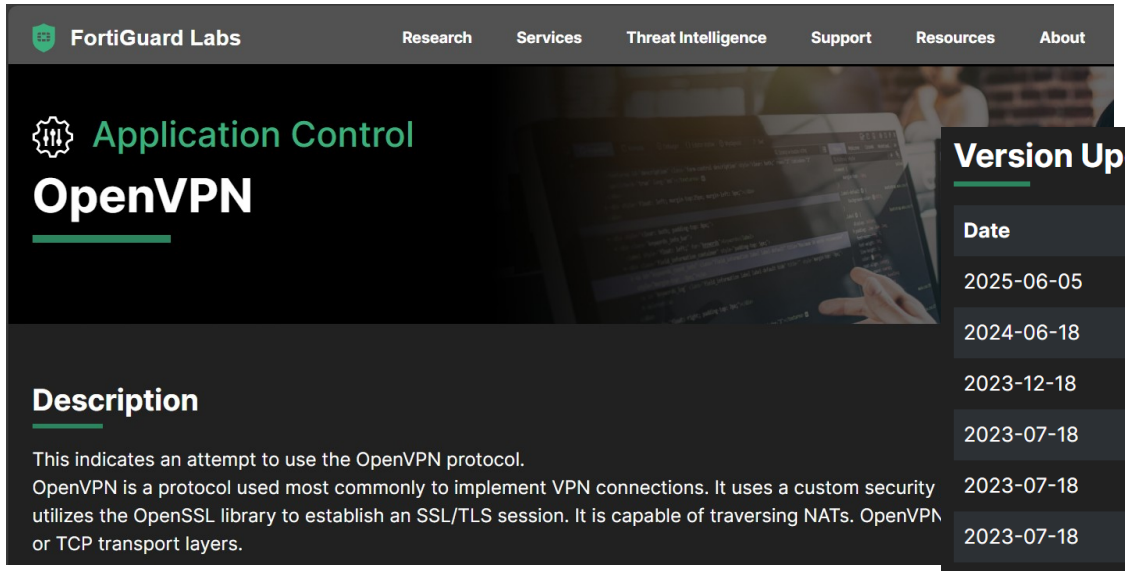
Firewall tested : Fortinet 2601F (FortiOS 7.6)

- Blocked vanilla OpenVPN
- XOR patch – bypassed the filter
  - (Not able to detected the dynamic change)

Firewalls

Static signatures try to keep up with the protocol changes





Version Updates

Date	Version	Detail
2025-06-05	33.020	Sig Added
2024-06-18	28.810	Sig Added
2023-12-18	26.698	Modified
2023-07-18	25.604	Sig Added
2023-07-18	25.603	Sig Added
2023-07-18	25.602	Sig Added
2023-05-09	23.549	Sig Added
2023-03-16	23.513	Sig Added
2023-03-07	23.507	Modified
2023-02-28	22.503	Modified

Fortinet Labs signature update [link], Juniper [link]

Firwalls

- Configurability
  - Limited to specific fields and fixed patterns
  - No dynamic Nature for detection for custom definitions
  - Ex- Fortinet custom rules

```
    edit "Block.Example.Custom"
        set signature "F-SBID( --attack_id 5150; --name "Block.example.com"; --pattern "example.com"; --
        protocol tcp; --service HTTP; --no_case; --flow from_client; --context host;)"
        set action block
        set comment ''
    next
end
```

VPN Detection Table

- Non-identifiable – No identifiable characteristics ○
- partially-identifiable – i.e., only 1 packet contains identifiable information ◔
- Identifiable -identifiable under certain conditions ◐
- Easily Identifiable –fixed across multiple packets ●

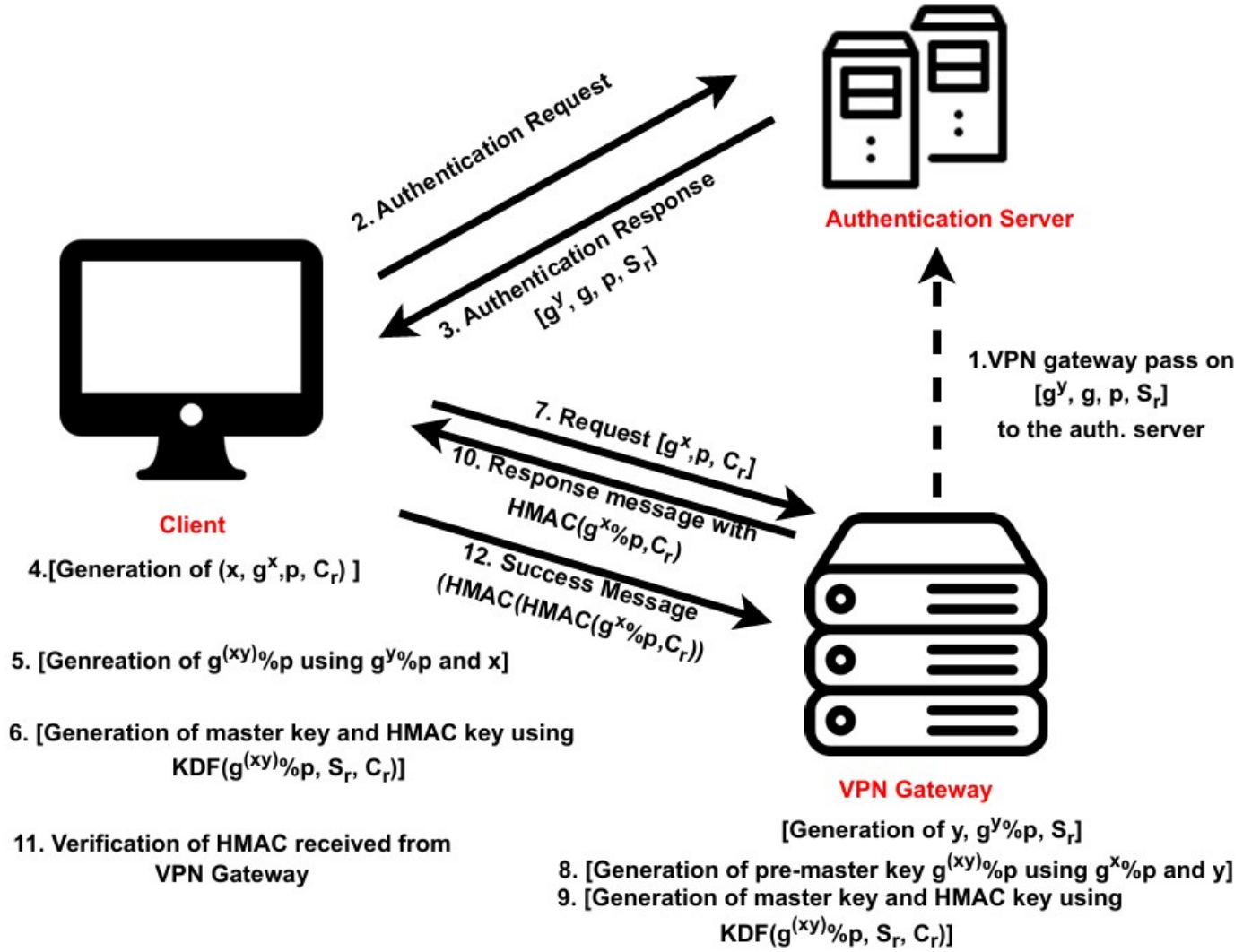
VPN Fingerprintability				
Protocol	**Type**	**Provider**	**Fingerprintable**	**Remarks**
OpenVPN (vanilla)	TCP UDP	NV,EV,PV	●	Control and data channel plaintext opcode patterns.
OpenVPN (Vanilla)+PSK	TCP UDP	None	○	Full packet encryption no visible patterns in traffic flow.
OpenVPN (Vanilla-XOR-based)	TCP UDP	NV,EV	◐	Control and data channel partially obfuscated using fixed XOR-string, but repeating sequence of apparently XOR-ed opcode at the same offset. Difficult to detect by commercial middleboxes like Fortinet 2600F series.
OpenVPN (Over Stunnel)	TCP	PP	○	Similar to the browser traffic.
OpenVPN (Over Obfsproxy)	TCP	PP	◔	No identifiable pattern but Obfsproxy can be identified [35].
OpenVPN (Over SSH)	TCP	PP	◐	Packet length sequence Identifiable
OpenVPN (Chameleon)	TCP	VP	○	Fully encrypted OpenVPN. flow with no identifiable features.
TLSv1.3 (Lightway)	TCP	EV	◔	Advertizes only a single cipher. suite in the TLS `ClientHello`.
TLSv1.3 (Stealth)	TCP	PV,WS	○	Similar to browser traffic with obscured SNI.
Websocket	TCP	WS	◔	SNI field and server name identifiable.
Wireguard (Vanilla)	UDP	VP,PV,WS	●	Fixed/repeating patterns in handshake packets.
Wireguard (NordLynx)	UDP	NV	●	Fixed/repeating patterns in handshake. packets.
Wireguard (WARP)	UDP	CF	●	Uses fixed/repeating pattern in Wireguard's `reserved` field.
UDP (Lightway)	UDP	EV	◔	Plaintext fingerprintable characteristic. in a single handshake message
IPsec/IKEv2	UDP	WS,VP,PP	○	High collateral damage if filtered.

Limitations

- Broader selection of VPN providers
- lack of access to ISP infrastructure - Inference based only on Wireshark captures and firewall hardware documentation
- Endpoint enumeration - Endpoint discovery was conducted via periodic cloud-based connections to VPN endpoints (spl. Public API requests)
- Cloudflare strictness - no SNI based evasion.
- Updates to application features - Findings are based on tested VPN versions and configurations; future updates may impact performance and detectability.

Our Proposed Fix

- Modification to OVPN pre-shared key mode
- Encryption of Complete packet payload
- Incorporating techniques for Authentication steps



Conclusion

- **Increasing Censorship** : Countries monitoring and banning VPNs to limit access to blocked content.
- **OpenVPN is vulnerable**: Prior research confirms OpenVPN flows and endpoints are detectable with specific configurations.
- **Alternative protocols offer resilience**: TLS, SSH, and IPSec are harder to detect and block without causing collateral damage.
- **Evasion through MTU manipulation**: Reducing MTU to fragment identifiable patterns across packets shows promise with minimal performance impact.
- **Future work**: Further research is needed to refine evasion strategies and assess long-term effectiveness against adaptive censors.

References

- [1] OpenVPN is Open to VPN Fingerprinting – Usenix'22 [link]
- [2] The Use of TLS in Censorship Circumvention – NDSS'19 [link]
- [3] Fortinet [link]
- [4] Juniper Firewalls Application Control [link]
- [5] TLS and SSH RFC [link1, link2]