

SharkFest'26 Europe

Detecting Covert Channels with Wireshark

Daniel Spiekermann

Sponsored by:
PROFI TAP DEEP NETWORK OBSERVABILITY
cPacket
TCP/IP training
endace Record. Respond.

About me

- Daniel Spiekermann

- Current position:
  - Professor for Distributed Systems at University of Applied Science and Arts in Dortmund, Germany (since 2023)
  - Founder and Research Lead at C²PANDA (Competence Center for Packet Acquisition and Network Data Analysis)
- Previous positions:
  - Professor for Digital Investigations at Police Academy of Lower Saxony, Germany (2020 - 2023)
  - Digital Investigator at Police Headquarter Dortmund, Germany (2013 – 2020)

… and some other IT jobs (mainly network and security) prior

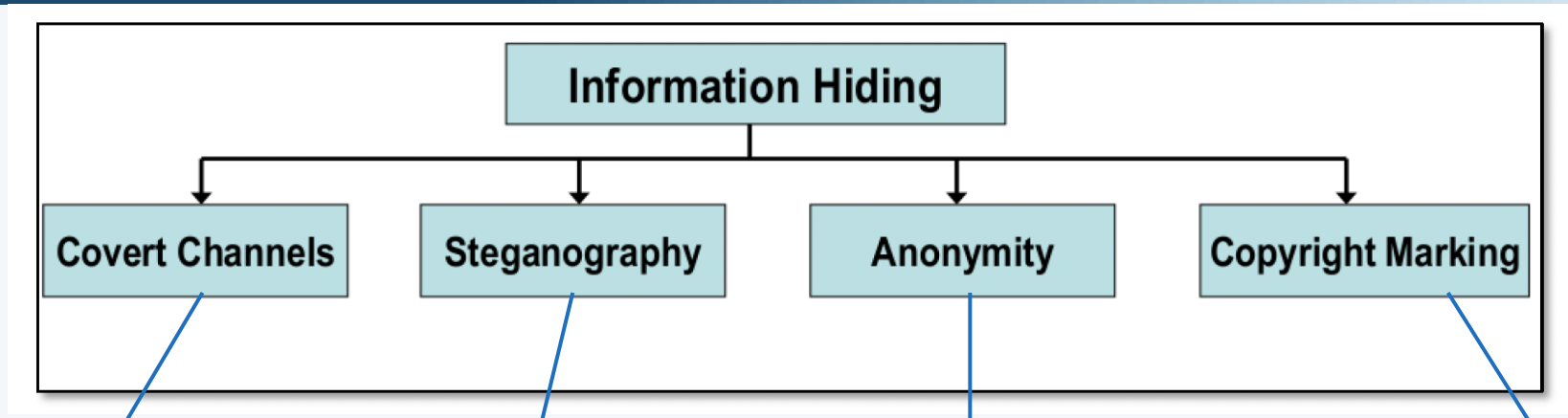
Overview

- Introduction
  - What is a covert channel?
  - Implementations
    - Storage-Based
    - Timing-Based
- Examples
- Countermeasures

Introduction

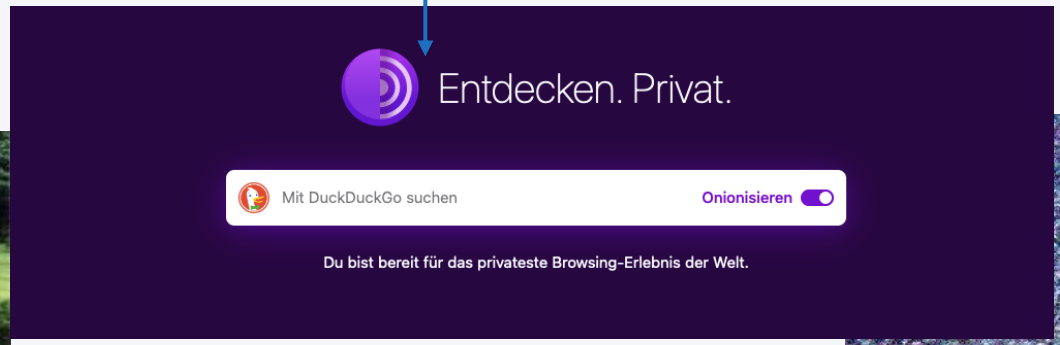


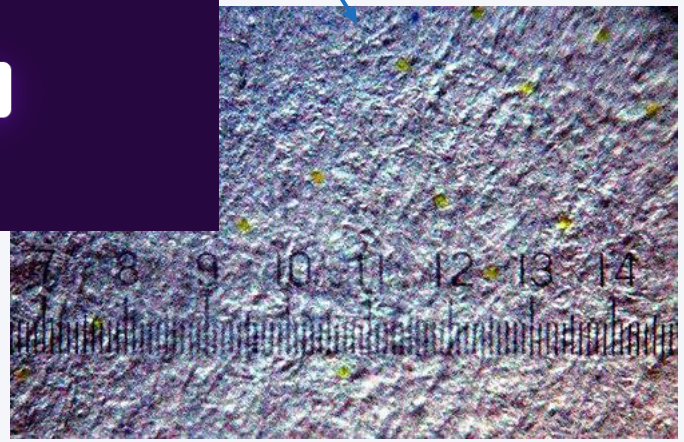
Overview



0.000042	192.168.10.100	85.13.131.35	TCP
0.000030	192.168.10.100	85.13.131.35	TCP
0.000020	192.168.10.100	85.13.131.35	TCP
0.000113	fe80::1005:5c7d:…	fe80::52e6:36ff:f…	ICMPv6
0.000303	192.168.10.100	85.13.131.35	TCP
0.000002	192.168.10.100	85.13.131.35	TCP
0.026233	85.13.131.35	192.168.10.100	TCP







What is a covert channel?

- Lampson[1] defines it as

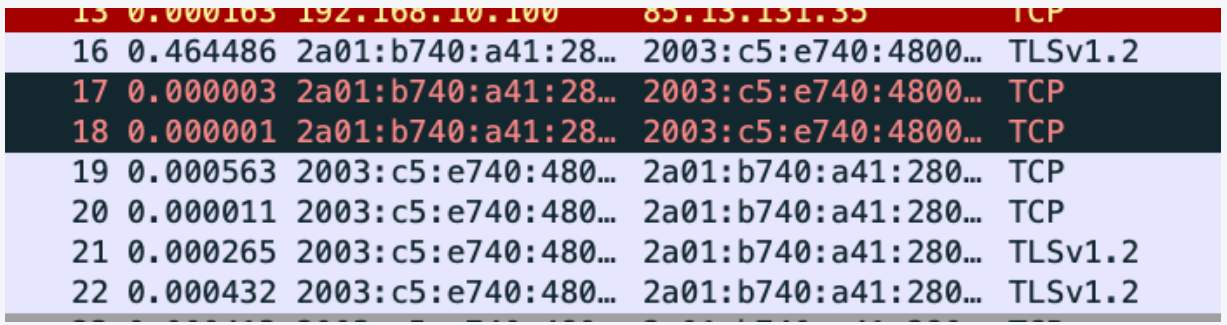
*Channels **not intended for information transfer at all**, such as the service program's effect on system load, to distinguish it from legitimate channels that are subjected to access controls by COMPUSEC.*

[1] Butler W. Lampson. „A note on the confinement problem". In: Commun. ACM 16.10 (Okt. 1973), S. 613–615. issn: 0001-0782. doi: 10.1145/362375.362389.

Comparison and differentiation

Cryptography

- Protects the content
- But communication is obvious

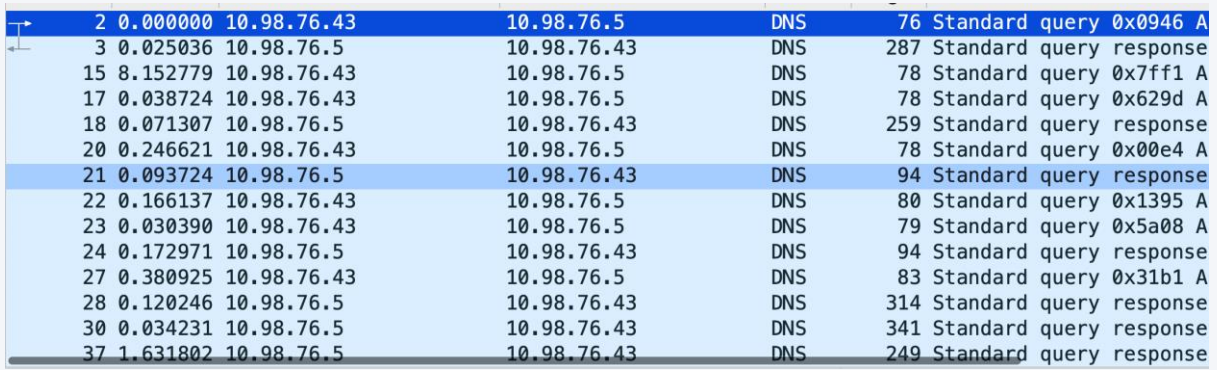


16	0.464486	2a01:b740:a41:28…	2003:c5:e740:4800…	TLSv1.2
17	0.000003	2a01:b740:a41:28…	2003:c5:e740:4800…	TCP
18	0.000001	2a01:b740:a41:28…	2003:c5:e740:4800…	TCP
19	0.000563	2003:c5:e740:480…	2a01:b740:a41:280…	TCP
20	0.000011	2003:c5:e740:480…	2a01:b740:a41:280…	TCP
21	0.000265	2003:c5:e740:480…	2a01:b740:a41:280…	TLSv1.2
22	0.000432	2003:c5:e740:480…	2a01:b740:a41:280…	TLSv1.2

- Combination is possible, aka hide encrypted data

Steganography

- Hides the existence of the content
- Communication is not obvious



2	0.000000	10.98.76.43	10.98.76.5	DNS	76 Standard query 0x0946 A
3	0.025036	10.98.76.5	10.98.76.43	DNS	287 Standard query response
15	8.152779	10.98.76.43	10.98.76.5	DNS	78 Standard query 0x7ff1 A
17	0.038724	10.98.76.43	10.98.76.5	DNS	78 Standard query 0x629d A
18	0.071307	10.98.76.5	10.98.76.43	DNS	259 Standard query response
20	0.246621	10.98.76.43	10.98.76.5	DNS	78 Standard query 0x00e4 A
21	0.093724	10.98.76.5	10.98.76.43	DNS	94 Standard query response
22	0.166137	10.98.76.43	10.98.76.5	DNS	80 Standard query 0x1395 A
23	0.030390	10.98.76.43	10.98.76.5	DNS	79 Standard query 0x5a08 A
24	0.172971	10.98.76.5	10.98.76.43	DNS	94 Standard query response
27	0.380925	10.98.76.43	10.98.76.5	DNS	83 Standard query 0x31b1 A
28	0.120246	10.98.76.5	10.98.76.43	DNS	314 Standard query response
30	0.034231	10.98.76.5	10.98.76.43	DNS	341 Standard query response
37	1.631802	10.98.76.5	10.98.76.43	DNS	249 Standard query response

History

- First occurence around 499 B.C. by Herodot:

ὁ γὰρ Ἰστιαῖος βουλόμενος τῷ Ἀρισταγόρῃ σημῆναι ἀποστῆναι ἄλλως μὲν οὐδαμῶς εἶχε ἀσφαλέως σημῆναι ὥστε φυλασσομένων τῶν ὁδῶν, ὁ δὲ τῶν δούλων τὸν πιστότατον ἀποξυρήσας τὴν κεφαλὴν ἔστιξε καὶ ἀνέμεινε ἀναφῦναι τὰς τρίχας· ὡς δὲ ἀνέφυσαν τάχιστα, ἀπέπεμπε ἐς Μίλητον ἐντειλάμενος αὐτῷ ἄλλο μὲν οὐδέν, ἐπεὰν δὲ ἀπίκηται ἐς Μίλητον, κελεύειν Ἀρισταγόρην ξυρήσαντά μιν τὰς τρίχας κατιδέσθαι ἐς τὴν κεφαλήν· τὰ δὲ στίγματα ἐσήμαινε, ὡς καὶ πρότερόν μοι εἴρηται, ἀπόστασιν

History - Translation

For Histiaeus, wishing to signal to Aristagoras that he should revolt, had no other safe way of sending the message, since the roads were guarded. So he **shaved the head of the most faithful of his slaves, marked it with tattooing, and waited for the hair to grow back**. As soon as it had grown back, he sent him to Miletus, giving him no other instruction than that, when he arrived in Miletus, he should tell Aristagoras to shave his hair and look at his head. And the marks signified, as I have said before, revolt.

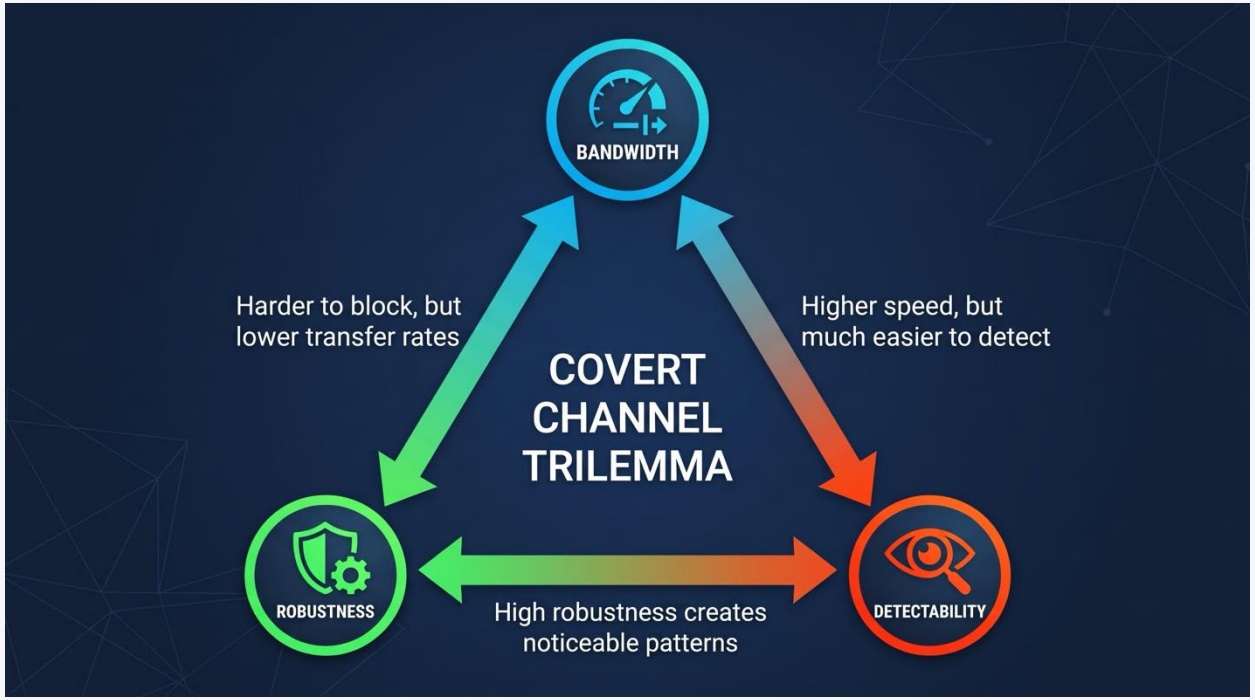
Types of (Network-based) Covert Channel

- Storage-Based Channels:
  - Protocols provide fields, options or „behaviour“, which can be used for covert channel
    - A flags field can transfer a single bit, which is analyzed by the receiver
    - Sequence numbers or retransmissions can be manipulated
    - Options (including padding)
    - Fields with undetermined content (identification) can transfer nearly every data
- Timing-Based Channels:
  - Network packets need time to travel from sender to receiver
    - Additional delay may indicate information
    - Longer or shorter intervals
    - Retransmissions
- Hybrid Channel:
  - Combination of both types

Defining the quality of a covert channel

- A covert channel is compared by three attributes:
  - Robustness: Resiliance against disturbances
  - Bandwidth: The amount of transferred data in a defined time
  - Detectability: Deviation from normal behavior

- Only two atrributes can be maximized
  - A channel with high bandwidth and high robustness is „easy“ to detect
  - A stealthy channel is unable to transfer lots of data in a short period of time



Overview of Network Covert Channel (not exhaustive)

- Network Covert Channels are available on nearly all layers:
  - Application Layer (HTTP [Bro+10], FTP [Zou+05], VoIP [Sae+20], NTP [Hie+21])
  - Transport Layer (TCP [ML05], UDP / Quic [Hua+25])
  - Network Layer (IPv4 [QSF04], IPv6 [MPC19], OSPF [SSK23])
  - Data Link Layer (ARP [JFM10], VXLAN [SKE17])
- Combinations are possible:
  - Protocol Hopping Covert Channels, presented in [WK12]
    - Data is hidden in different protocols (e.g. HTTP → ICMP → DNS)
- Research:
  - DYST: An Amplified Covert Channel That Points To Previously Seen Data [Wen+25]
  - Indirect Covert Channel in OpenFlow [SK25]

Let‘s use Wireshark

Example 1

Example 1

- Uses ICMP Payload in Echo request (ICMP Type 8, Code 0)
- „Not really“ defined in RFC 792

Description: The data received in the echo message must be returned in the echo reply message.

- Value is valid, so Wireshark does not detect an issue

```
Internet Control Message Protocol
  Type: Echo (ping) reply (0)
  Code: 0
  Checksum: 0x1878 [correct]
  [Checksum Status: Good]
  Identifier (BE): 0 (0x0000)
  Identifier (LE): 0 (0x0000)
  Sequence Number (BE): 0 (0x0000)
  Sequence Number (LE): 0 (0x0000)
  [Request frame: 3]
  [Response time: 0.036 ms]
> Data (31 bytes)
    Data: 434f564552545f444154415f4348554e4b3a5b435d5f50414444494e475f58
    [Length: 31]
```

```
bc 24 11 fc 66 21 bc 24  11 76 be d1 08 00 45 00   ·$··f!·$ ·v····E·
00 3b 7e 9b 00 00 40 01  66 75 c0 a8 0a 30 c0 a8   ·;~···@· fu···0··
0a 31 00 00 18 78 00 00  00 00 43 4f 56 45 52 54   ·1···x·· ··COVERT
5f 44 41 54 41 5f 43 48  55 4e 4b 3a 5b 43 5d 5f   _DATA_CH UNK:[C]_
50 41 44 44 49 4e 47 5f  58                       PADDING_ X
```

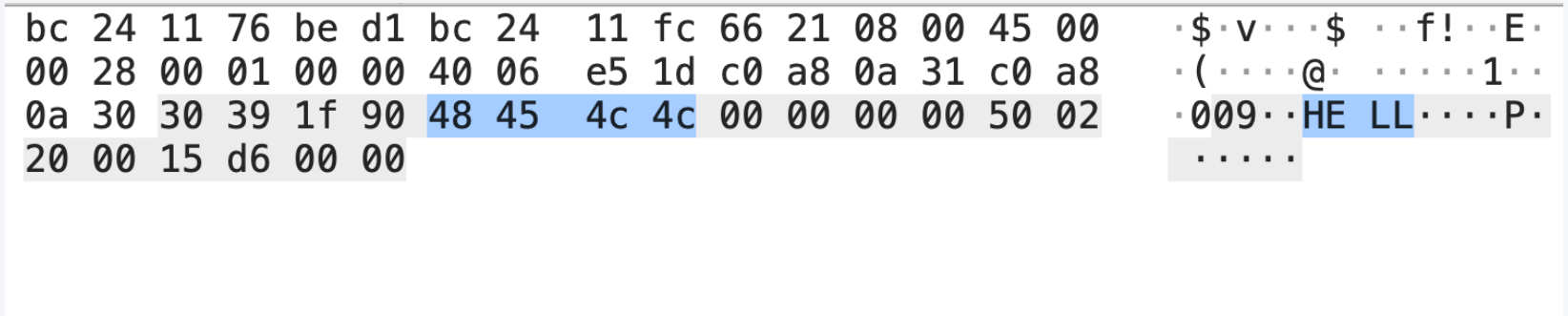
Example 2

Example 2

- Manipulates Initial Sequence Number (ISN) in TCP-Header
- RFC 9293

> TCP initial sequence numbers are generated from a number sequence that monotonically increases until it wraps, known loosely as a "clock". This clock is a 32-bit counter that typically increments at least once every roughly 4 microseconds, although it is neither assumed to be realtime nor precise, and need not persist across reboots. The clock component is intended to ensure that with a Maximum Segment Lifetime (MSL), generated ISNs will be unique since it cycles approximately every 4.55 hours, which is much longer than the MSL. Please note that for modern networks that

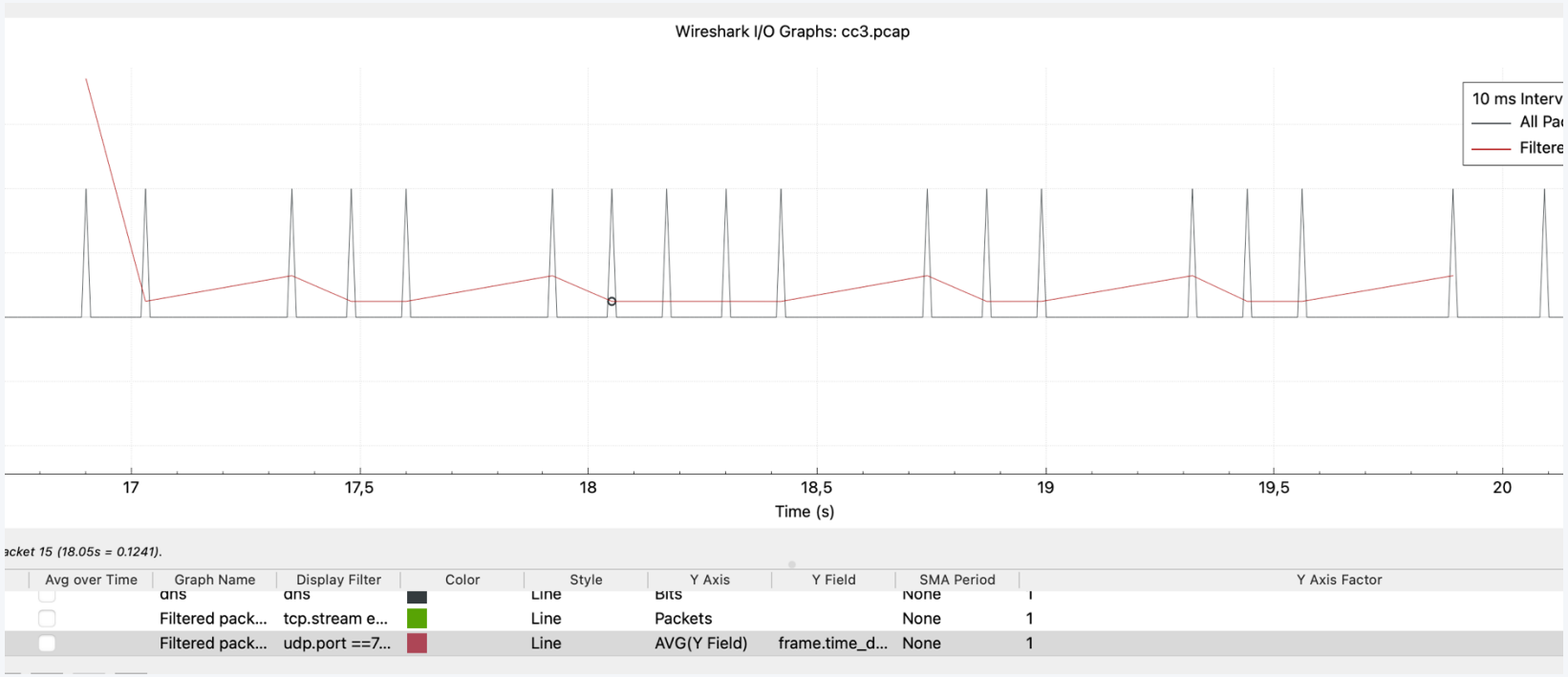
- Closed port used, but open ports are possible
- Wireshark shows no indication of a problem (ISNs are valid)



Example 3

Example 3

- Used packet delay between (UDP port 7777) transmitted packets (= timing-based covert channel)
- No direct detection possible
- „Detectable“ via I/O Graph
- Without knowing the encoding the translation is nearly impossible



01001000 01001001
= 72 73
= H I

Example 4

Example 4

- Uses IPv4 ID field
- Defined in **RFC 6864**

The IPv4 Identification (ID) field enables fragmentation and reassembly and, as currently specified, is required to be unique within the maximum lifetime for all datagrams with a given source address/destination address/protocol tuple.

- Value is valid, so Wireshark does not detect any issue
- Information is „covert“ with ASCII-Code (in Hex)
  - 0x0053 = 83 = S
  - 0x0045 = 69 = E
  - 0x0043 = 67 = C
  - 0x0052 = 82 = R
  - 0x0045 = 69 = E
  - 0x0054 = 84 = T
  - …

Example 4

- Uses IPv4 ID field
- Defined in RFC 6864

The IPv4 Identification (ID) field enables fragmentation and reassembly and, as currently specified, is required to be unique within the maximum lifetime for all datagrams with a given source address/destination address/protocol tuple.

- Value is valid, so Wireshark does not detect any issue

```
7.  Security Considerations

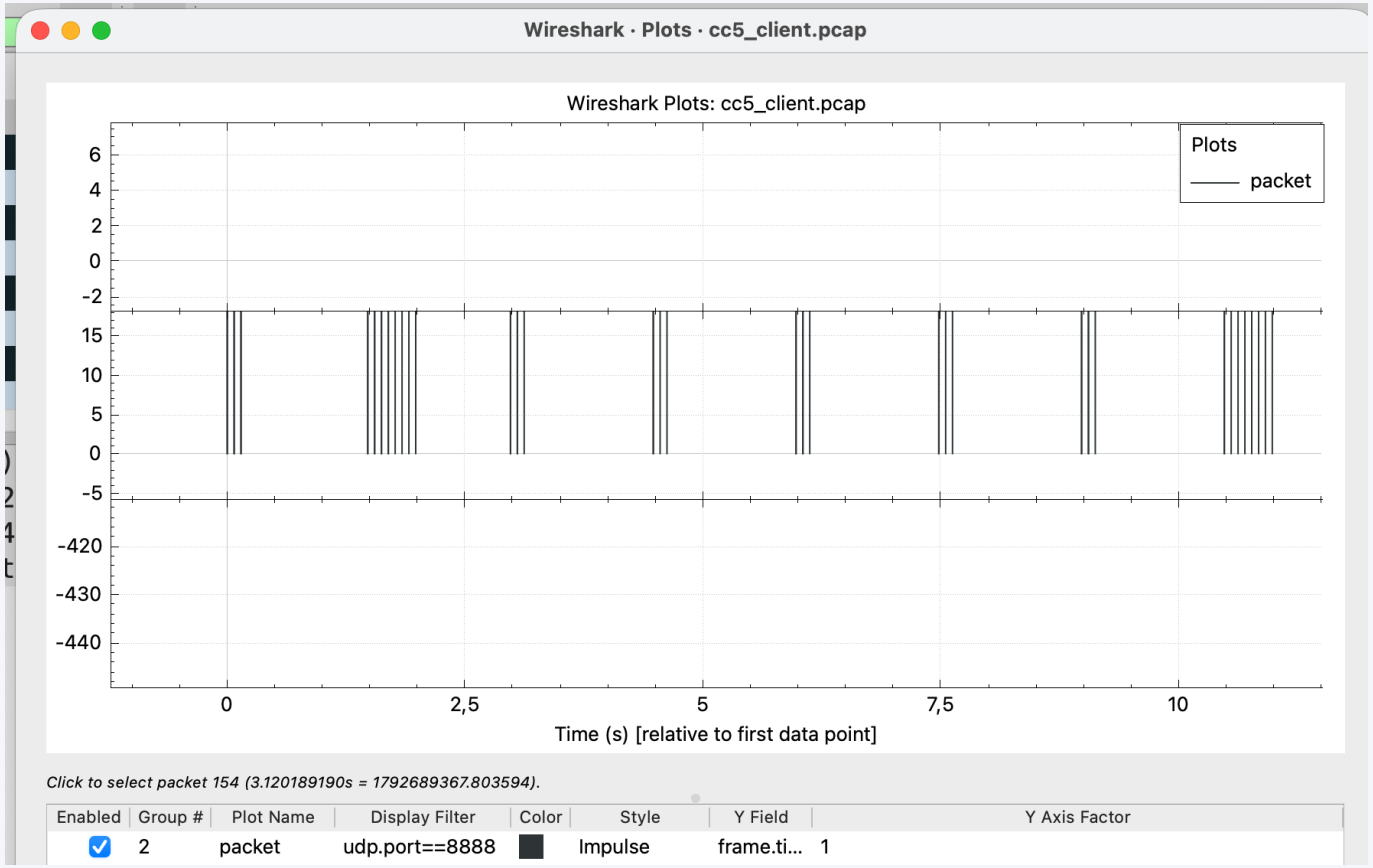
   When the IPv4 ID is ignored on receipt (e.g., for atomic datagrams),
   its value becomes unconstrained; therefore, that field can more
   easily be used as a covert channel.  For some atomic datagrams it is
   now possible, and may be desirable, to rewrite the IPv4 ID field to
   avoid its use as such a channel.  Rewriting would be prohibited for
   datagrams protected by the IPsec Authentication Header (AH), although
   we do not recommend use of the AH to achieve this result [RFC4302].
```

Example 5

Example 5

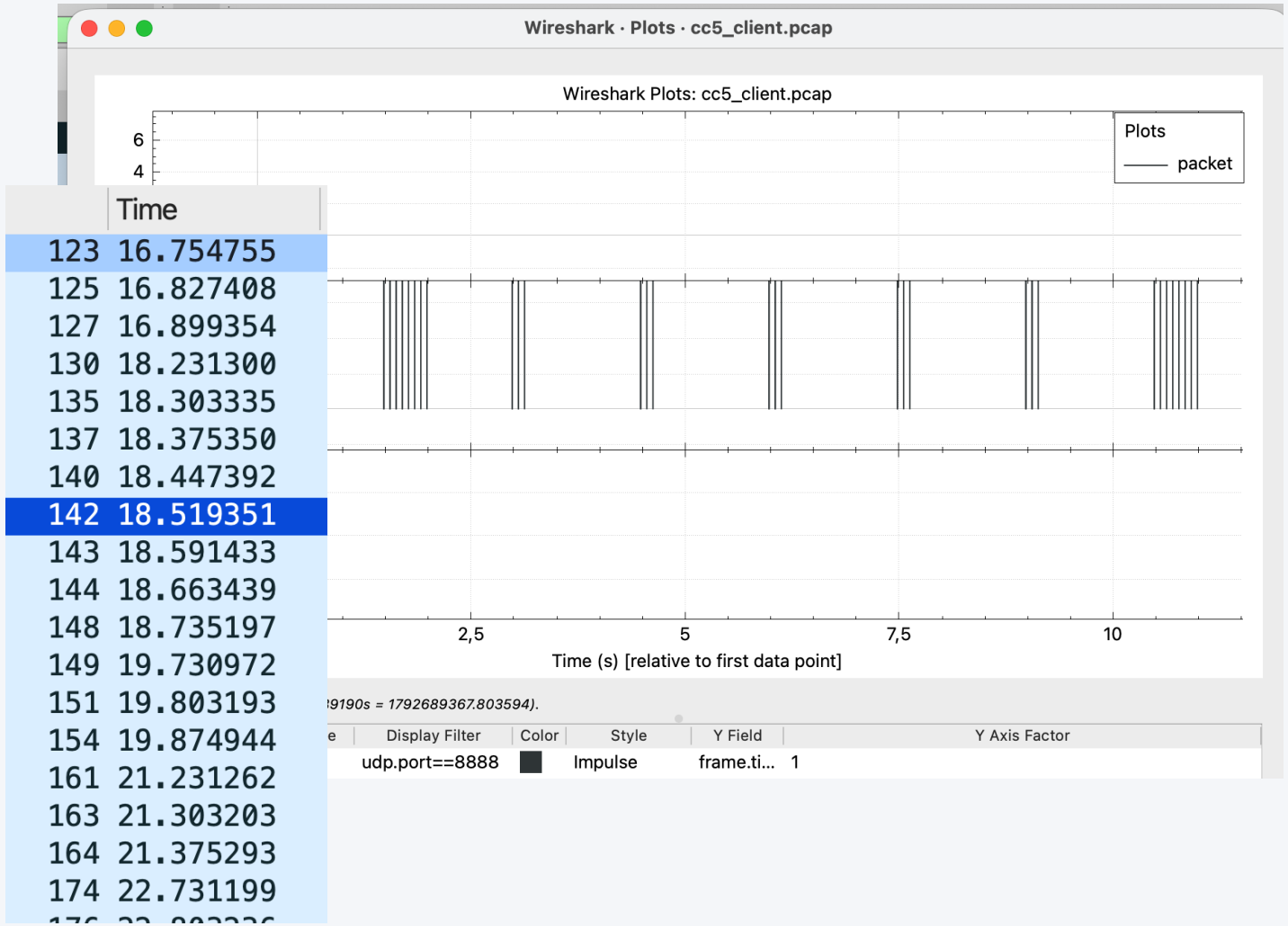
- Sends bursts of traffic and encodes the information
  - 3 packet-burst defines 0
  - 8 packet-burst defines 1
  - 01000001 = A

- „Detectable“ via I/O Graph or via Plots



Example 5

- Sends bursts of traffic and encodes the information
  - 3 packet-burst defines 0
  - 8 packet-burst defines 1
  - 01000001 = A
- „Detectable“ via I/O Graph or via Plots
- Additional detection:
  - View -> Time Display Format
  - Seconds since first captured packet



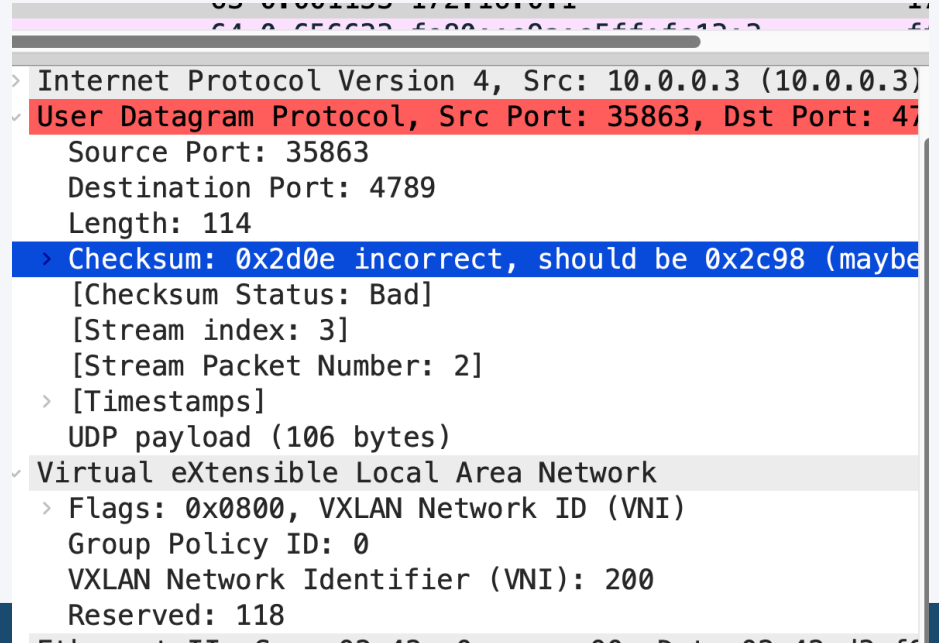
Example 6

Example 6

- Uses VXLAN RSVD2 field
  - RSVD1 is even possible, but is (sometimes) used as group policy descriptor
  - RFC 7348 - VXLAN, Sec. 5 VXLAN Frame Format

> – **Reserved fields (24 bits and 8 bits): MUST be set to zero on transmission and ignored on receipt.**

- Wireshark detects an issue, but does not directly point to the correct protocol
  - UDP instead of VXLAN
  - But the message is ok, the checksum is incorrect



Example 7

Example 7

- HTTP-based covert channel
- Adds HTTP-header „X-Session-Data“
  - User-defined header, but not valid since 2012 (RFC 6648 - Deprecating the "X-" Prefix and Similar Constructs in Application Protocols) any more (X-header)
  - Use is still accepted by multiple server
- Uses Base64 to hide plain text

```
> echo cHc9djNyeVMzQ3VyMw== |base64 --decode
pw=v3ryS3Cur3
```

```
› Frame 4: Packet, 229 bytes on wire (1832 bits), 229 bytes captured (1
› Ethernet II, Src: 00:11:22:33:44:55, Dst: 66:77:88:99:aa:bb
› Internet Protocol Version 4, Src: 192.168.1.50 (192.168.1.50), Dst: 1
› Transmission Control Protocol, Src Port: 49152, Dst Port: 80, Seq: 1,
⌄ Hypertext Transfer Protocol
  › GET /index.html HTTP/1.1\r\n
    Host: test.frnsc.de\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64)\r\n
    Accept: text/html\r\n
    X-Session-Data: cHc9djNyeVMzQ3VyMw==\r\n
    Connection: keep-alive\r\n
    \r\n
    [Response in frame: 5]
    [Full request URI: http://test.frnsc.de/index.html]
```

Example 8

Example 8

- Timestamp looks unsuspicious
- Uses NTP LSB of Transmit Timestamp
- 2026-10-24T12:18:05.639608707 = ee 87 20 7d a3 bd 65 73
- An NTP-timestamp is divided in two big-endian parts:
  - Seconds (4 Bytes / 32 Bit): ee 87 20 7d
  - Microseconds (4 Bytes / 32 Bit): a3 bd 65 73
  - Conversion to Unix-Timestamp
- Again encoding of ASCII:
  - 65 = e
  - 73 = s

```
> User Datagram Protocol, Src Port: 12345, Dst Port: 123
v Network Time Protocol (NTP Version 4, client)
  > Flags: 0x23, Leap Indicator: no warning, Version number
    [Response In: 197]
    Peer Clock Stratum: primary reference (1)
    Peer Polling Interval: 10 (1024 seconds)
    Peer Clock Precision: -20 (0.000000954 seconds)
    Root Delay: 0.000000 seconds
    Root Dispersion: 0.000992 seconds
    Reference ID: uncalibrated local clock
    Reference Timestamp: Oct 24, 2026 12:18:04.639602661 UT
    Origin Timestamp: NULL
    Receive Timestamp: NULL
    Transmit Timestamp: Oct 24, 2026 12:18:05.639608707 UTC
```

```
50 e6 36 d7 f3 7c bc 24  11 fc 66 21 08 00 45 00   P·6··|·$ ··f!··E·
00 4c 00 01 00 00 40 11  e5 1d c0 a8 0a 31 c0 a8   ·L····@· ·····1··
0a 01 30 39 00 7b 00 38  b0 85 23 01 0a ec 00 00   ··09·{·8 ··#·····
00 00 00 00 00 41 4c 4f  43 4c ee 87 20 7c a3 bd   ·····ALO CL·· |··
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ···· ···· ····
00 00 ee 87 20 7d a3 bd  65 73                     ···· }·· es
```

Countermeasures



Common Countermeasuares

- Active and passive warden
  - Monitor network traffic and detect or eliminate covert information
- Traffic Normalization
  - Set reserved fields back to „0“ or default values
  - Add or remove packet interval times
- Traffic Analysis and Anomaly Detection
  - Identify unusual traffic patterns, timing deviations, and protocol-field distributions
- Protocol Filtering and Egress Control
  - Restrict unnecessary protocols and block unauthorized communication paths
- Secure Implementation
  - Understand known covert-channel techniques and design protocols and software to minimize exploitable fields and behaviors

First steps with Wireshark

- Check Expert Info
  - Display filter: `_ws.expert` or `_ws.expert.severity matches "Warning|Error"`)

> **Expert information is only a hint**
>
> Expert information is the starting point for investigation, not the stopping point. Every network is different, and it's up to you to verify that Wireshark's expert information applies to your particular situation. The presence of expert information doesn't necessarily indicate a problem and absence of expert information doesn't necessarily mean everything is OK.

- Use graphs or plots for timing-based covert channel
- Use different setups of Time Display Format
- Take a look into packet bytes pane (Information might be „decoded“ automatically)

Let‘s sum it up



Real World Example 1

New SteganoAmor attacks use steganography to target 320 orgs globally

By **Bill Toulas**

April 15, 2024 04:31 PM 0

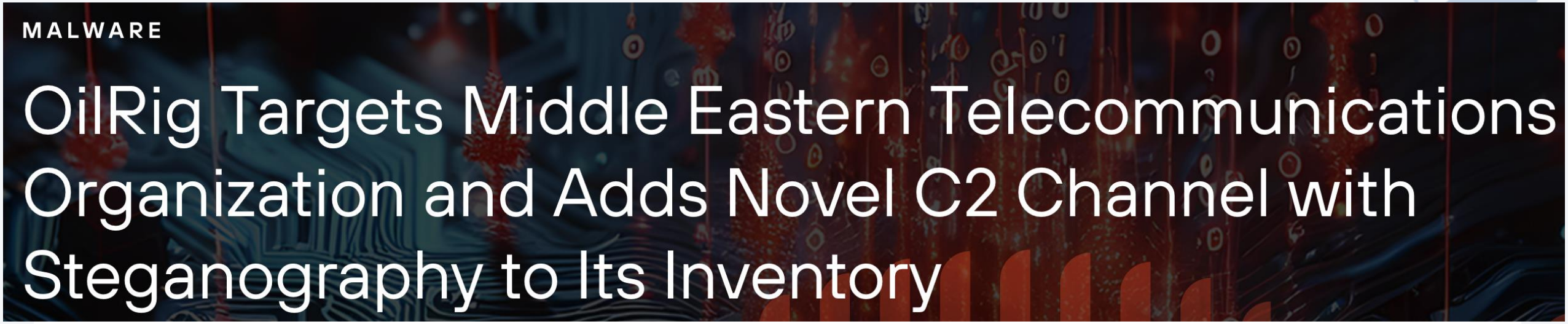
PowerShell code inside the script contained in the image downloads the final payload hidden inside a text file in the form of a reversed base64-encoded executable.

```
GET /romamammamamamaa.txt HTTP/1.1
Host: 23.95.60.74
Connection: Keep-Alive

HTTP/1.1 200 OK
Content-Type: text/plain
```

https://www.bleepingcomputer.com/news/security/new-steganoamor-attacks-use-steganography-to-target-320-orgs-globally

Real World Example 2



RDAT Backdoor

The adversaries compiled the RDAT payloads used in the attacks on the Middle Eastern telecommunications organization on March 1, 2020, and configured it to use a domain provided on the command line or the hardcoded domain `rsshay[.]com` as its C2 server. Unlike previous RDAT samples, this particular sample only uses DNS tunneling for its C2 communications with no HTTP fallback channel. This RDAT sample can only use TXT queries in its DNS tunnel and will issue queries structured like the following:

https://unit42.paloaltonetworks.com/oilrig-novel-c2-channel-steganography

Conclusion

Covert channel pose a real threat in modern networks

- Wireshark is perfect for in-depth analysis of packets
- But the number of packets in a typical capture is too high too find the needle in the haystack without further information
  - Pre-Filtering
  - Combination of techniques (e.g. flow-based approaches)
  - Anomaly detection

One last thing

"If you think you are protected against a covert channel…

you just haven't detected one."

Did you see it?

There is a hidden message (a word) in this presentation.
If you spot it and can tell me what it is, I'll buy you a coffee!

Thank You!



Contact:
Daniel Spiekermann
Discord: lo_127001
daniel.spiekermann@fh-dortmund.de

References

[Bro+10] Erik Brown u. a. „Covert channels in the HTTP network protocol: Channel characterization and detecting Man-in-the-Middle attacks". In: Journal of Information Warfare 9.3 (2010), S. 26–38.

[Cav21] Luca Caviglione. „Trends and Challenges in Network Covert Channels Countermeasures". In: Applied Sciences 11.4 (2021). issn: 2076-3417. doi: 10.3390/app11041641.

[Hal25] Mike Hall. GitHub - mchall/HiddenWatermark: Embeds a hidden watermark in an image using a blind DWT-DCT approach — github.com. https://github.com/mchall/HiddenWatermark?tab=readme-ov-file. [Accessed 05-09-2025]. 2025.

[Hie+21] Jonas Hielscher u. a. „A systematic analysis of covert channels in the networktime protocol". In: Proceedings of the 16th International Conference on Availability,Reliability and Security. 2021, S. 1–11. 42 /

[Hua+25] Jianan Huang u. a. „QuicCourier: Leveraging the Dynamics of QUIC-Based WebsiteBrowsing Behaviors Through Proxy for Covert Communication". In: IEEE Transactionson Dependable and Secure Computing (2025).

[JFM10] Liping Ji, Yu Fan und Chuan Ma. „Covert channel for local area network". In:2010 IEEE International Conference on Wireless Communications, Networking andInformation Security. IEEE. 2010, S. 316–319.

[Lam73] Butler W. Lampson. „A note on the confinement problem". In: Commun. ACM 16.10(Okt. 1973), S. 613–615. issn: 0001-0782. doi: 10.1145/362375.362389.43 / 46

[ML05] Steven J Murdoch und Stephen Lewis. „Embedding covert channels into TCP/IP". In: International Workshop on Information Hiding. Springer. 2005, S. 247–261.

[MPC19] Wojciech Mazurczyk, Krystian Pow.jski und Luca Caviglione. „IPv6 covert channelsin the wild". In: Proceedings of the third central european cybersecurity conference.2019, S. 1–6.

[QSF04] Haipeng Qu, Purui Su und Dengguo Feng. „A typical noisy covert channel in theIP protocol". In: 38th Annual 2004 International Carnahan Conference on SecurityTechnology, 2004. IEEE. 2004, S. 189–192.

[Sae+20] Jens Saenger u. a. „VoIP network covert channels to enhance privacy and informationsharing". In: Future Generation Computer Systems 111 (2020), S. 96–106.44 / 46

[Sim84] Gustavus J. Simmons. „The Prisoners' Problem and the Subliminal Channel". In:Advances in Cryptology – CRYPTO '83. Plenum, 1984, S. 51–67.

[SK25] Daniel Spiekermann und Jörg Keller. „An Indirect RTT-Based Covert Channel inOpenFlow-Enabled SDN Networks". In: 2025 IEEE Conference on Network FunctionVirtualization and Software-Defined Networking (NFV-SDN). IEEE. 2025, S. 1–6.

[SKE17] Daniel Spiekermann, Jörg Keller und Tobias Eggendorfer. „Towards covert channels incloud environments: a study of implementations in virtual networks". In: InternationalWorkshop on Digital Watermarking. Springer. 2017, S. 248–262.

[SSK23] Michael Schneider, Daniel Spiekermann und J.rg Keller. „Network covert channels in routing protocols". In: Proceedings of the 18th International Conference on Availability, Reliability and Security. 2023, S. 1–8.

[Wen+25] Steffen Wendzel u. a. „ DYST (Did You See That?): An Amplified Covert ChannelThat Points To Previously Seen Data ". In: IEEE Transactions on Dependable and Secure Computing 22.01 (Jan. 2025), S. 614–631. issn: 1941-0018. doi: 10.1109/TDSC.2024.3410679.

[WK12] Steffen Wendzel und Jörg Keller. „Design and implementation of an active warden addressing protocol switching covert channels". In: Proc. 7th International Conference on Internet Monitoring and Protection (ICIMP 2012). IARIA. 2012, S. 1–6.

[Zou+05] Xin-guang Zou u. a. „The research on information hiding based on command sequenceof FTP protocol". In: International Conference on Knowledge-Based and Intelligent Information and Engineering Systems. Springer. 2005, S. 1079–1085.