

PiLens

network forensics for the Raspberry Pi



The friendly field guide

Turn a Raspberry Pi into a Wi-Fi test network you can actually *see* inside.

→ TL;DR

This guide is built for skimming. Every section opens with a **TL;DR**. Boxes in the margins and colour-coded callouts carry the important bits. **Read the box, skip the prose, come back if you need it.** You will not miss anything vital by jumping around — that is the point.

How to read this (30 seconds)

- TL;DR

The one-line version. If you read nothing else, read these.
- 🔊

Heads-up

Something that will bite you if you ignore it.
- 👁

Pro tip

A shortcut or nicety. Safe to skip.
- 📖

Honest truth

Where PiLens tells you what it *cannot* do. Read these — they save hours of confusion.

👁

Pro tip

In a hurry? Jump straight to **§1 The ten-minute start**, connect a phone, and open **Live Capture**. Come back for the rest when you are curious.

Contents

How to read this	1
1 The ten-minute start	2
1.1 What you need	2
1.2 The steps	2
2 What PiLens is (and isn't)	3
2.1 The two padlocks	3
3 A tour of the screens	4
3.1 Live Capture — the main event	4
3.2 Filters — find the needle	5
3.3 Conversations & Follow TCP Stream	5
3.4 DNS Explorer	6
3.5 TLS Explorer	6
3.6 Statistics	7
3.7 Findings	8
3.8 HTTPS inspection (Mode B) — the opt-in one	9
3.9 The housekeeping screens	11
4 Common tasks, as recipes	12
4.1 “See everything one device does”	12
4.2 “Save this capture as evidence”	12
4.3 “Analyse a capture from somewhere else”	12
4.4 “Stop a device reaching the internet but still watch it”	12
5 When something's off	13
6 Keyboard cheat sheet	13

1 The ten-minute start

→ TL;DR

Flash the SD card > plug in **Ethernet** > run one installer > open the web page > finish the wizard > connect a test device to the new Wi-Fi > watch its traffic. That is the whole dance.

What you need

- A Raspberry Pi **4 or 5** running Raspberry Pi OS Lite (64-bit).
- An **Ethernet** cable to your normal network.
- The Pi's built-in Wi-Fi (this becomes the test network).
- A phone or laptop you own, to be the "device under test".

You manage
PiLens over
wire, never
its own Wi-

The steps

1. Copy the release onto the Pi and run `sudo ./install.sh`. It explains everything before it touches anything, backs up your config, and **rolls back if anything fails**.
2. It prints three things: a **web address**, a **certificate fingerprint**, and a one-time **setup token**. Keep them.
3. Open the web address from a computer on your wired network. Your browser will warn about the certificate — that is expected (see §2). Check the fingerprint matches, then continue.
4. Work through the **setup wizard**: create your login, choose the management network, name the Wi-Fi, pick a password, done.
5. Connect your test device to the new Wi-Fi network.
6. Open **Live Capture**, press **Start capture** on the Wi-Fi interface, and watch the packets roll in.

Heads-up

The moment you create the access point, the Pi's own Wi-Fi *becomes* that access point. If you were connected to the Pi over Wi-Fi you will be kicked off. **Always manage it over Ethernet.**

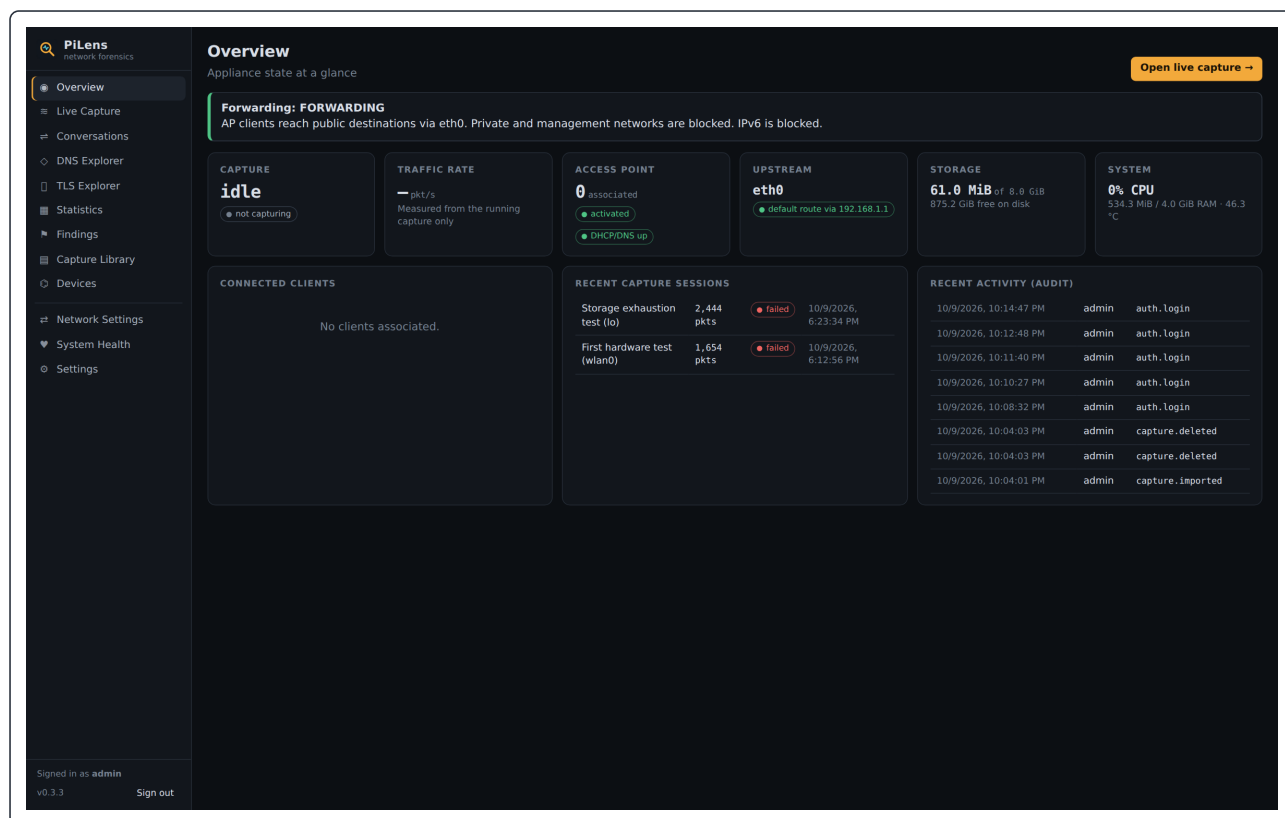


Figure: The Overview dashboard — appliance status, forwarding state, clients, and a live glance at everything at once.

2 What PiLens is (and isn't)

→ TL;DR

PiLens is a **router with X-ray vision**. Devices on its Wi-Fi go through it to reach the internet, and it records every packet. It shows you *metadata and packets* — it does **not** decrypt HTTPS.

PiLens sits between a test device and the internet. The device joins its Wi-Fi, gets an address, and routes out through the Pi's Ethernet. Because *every* packet crosses the Pi, PiLens can capture and analyse all of it — addresses, ports, DNS, TLS handshakes, timing, the lot.

The honest truth

Encrypted means encrypted. HTTPS, QUIC, VPNs and encrypted DNS stay scrambled. PiLens shows you *who* talked to *whom*, *when*, over *what* — the server name, the TLS version, the certificate when it is sent in the clear — but **not the contents**. That is TLS doing its job, not PiLens failing. Decryption needs a separate, explicit step on each device (Mode B, §3.9), and **a trusted certificate can never be pushed down the network** — the device's owner must install it by hand. Anything that claimed otherwise would be lying to you.

The two padlocks

- **The management padlock** is the HTTPS on the web interface. It is self-signed, so your browser warns once. Check the fingerprint the installer printed, trust it, move on.
- **The interception padlock** (decrypting a device's HTTPS) is a *separate*, opt-in, per-device thing: a toggle on the Certificates page. See §3.9.

3 A tour of the screens

→ TL;DR

Left-hand menu, top to bottom: **Overview** (glance), **Live Capture** (the main event), **Conversations** / **DNS** / **TLS** / **Statistics** / **Findings** (ways to slice what you caught), then **Library**, **Devices**, **Network**, **Health**, **Settings** (housekeeping).

Live Capture — the main event

→ TL;DR

Three stacked panes, exactly like Wireshark: the **packet list** up top, the **protocol tree** in the middle, the **raw bytes** at the bottom. Click a packet; the other two follow.

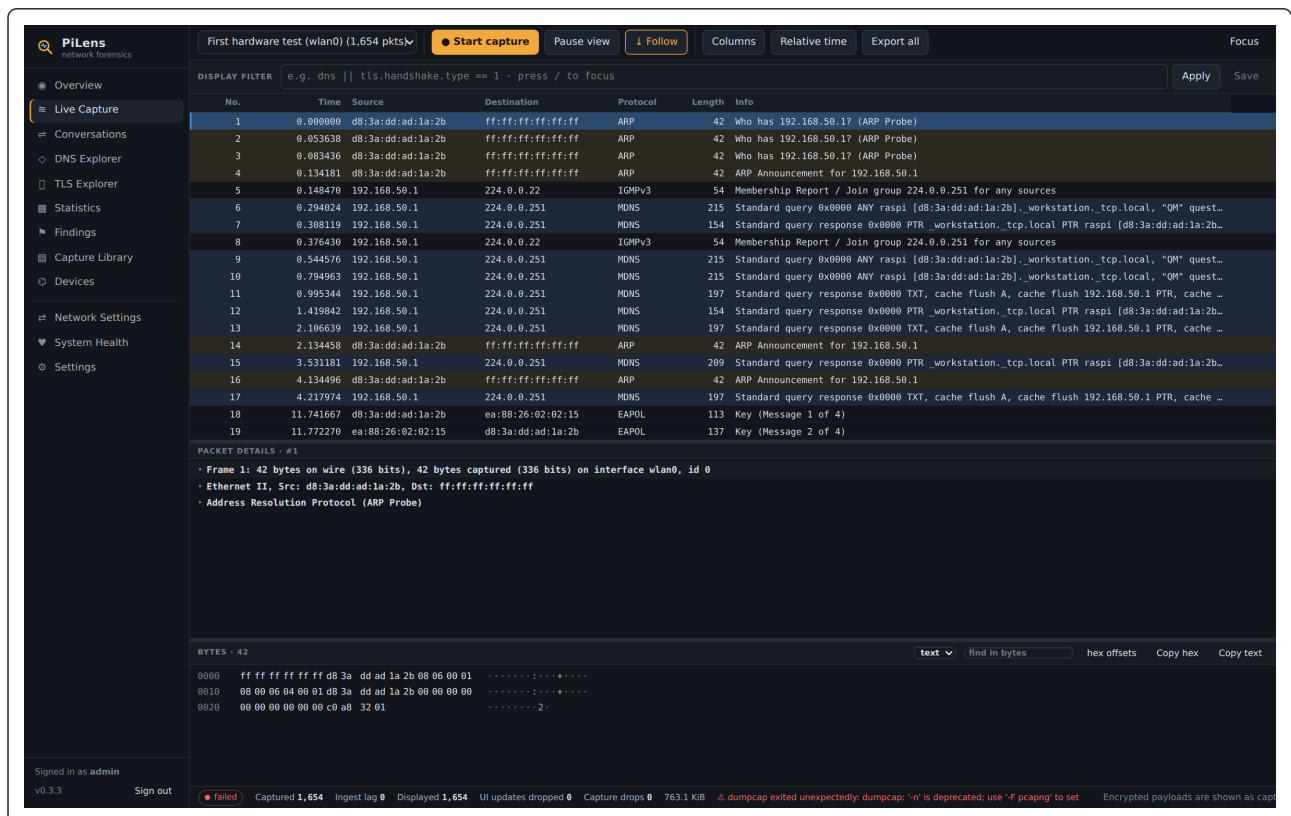


Figure: Live Capture. Packet list, protocol tree and a hex/ASCII inspector. The status bar along the bottom keeps you honest.

The packet list streams live and is colour-coded by protocol. It stays smooth even on a busy network because only the visible rows are drawn.

The protocol tree shows the selected packet decoded field by field, by TShark (the Wireshark engine). Click a field and the matching bytes light up below — but *only* when PiLens can prove those bytes really are that field. It will never make up a highlight.

The byte pane is a proper hex inspector: offsets, hex, ASCII, search, copy-as-hex or text.

👉 Pro tip

The bottom status bar separates things people usually confuse: **Captured** (packets stored), **Ingest lag** (stored but not yet summarised), **UI updates dropped** (your browser fell behind — harmless), and **Capture drops** (the appliance genuinely missed packets — not harmless). A dropped *screen update* is not a dropped *packet*.

🔔 Heads-up

Pause view and **Pause capture** are different buttons. “Pause view” just freezes the scrolling so you can read; capture keeps running. “Pause capture” actually stops storing packets. “Stop” ends the session for good.

Filters — find the needle

→ TL;DR

Type a **Wireshark display filter** in the bar (e.g. `dns`, `tcp.port == 443`, `ip.addr == 192.168.50.20`) and press Enter. It is evaluated by TShark, so real Wireshark syntax works.

The box turns green when the filter is valid, red when it isn't, with the reason. Press **/** anywhere to jump to it. Right-click a packet or a field to build a filter without typing.

Conversations & Follow TCP Stream

→ TL;DR

Conversations groups traffic by endpoint pair with bytes each way. Hit **Follow** on a TCP row to reassemble the whole back-and-forth.

The screenshot shows the PiLens interface with the 'Conversations' tab selected. The table displays traffic grouped by endpoint pairs, with columns for Endpoint A, Endpoint B, Protocols, Packets, Bytes, and Duration. The first row is highlighted, showing a conversation between 17.57.146.154:5223 and 192.168.50.101:60686, with 579 packets and 370.2 KiB of data. The table is sortable, and each row has a 'Follow' button and a 'Packets' link.

ENDPOINT A	ENDPOINT B	PROTOCOLS	PACKETS	BYTES	A→B	B→A	DURATION	
17.57.146.154:5223	192.168.50.101:60686	SSL TCP TLS	579	370.2 KiB	109.2 KiB	260.9 KiB	1m 3s	Follow Packets
17.171.47.23:443	192.168.50.101:60696	SSL TCP TLS	33	17.2 KiB	11.3 KiB	6.0 KiB	2s	Follow Packets
17.250.81.26:443	192.168.50.101:60689	SSL TCP TLS	38	14.5 KiB	7.7 KiB	6.8 KiB	3s	Follow Packets
17.250.81.18:443	192.168.50.101:60698	SSL TCP TLS	36	14.5 KiB	8.5 KiB	6.1 KiB	2s	Follow Packets
17.250.81.69:443	192.168.50.101:60693	SSL TCP TLS	39	12.7 KiB	7.3 KiB	5.4 KiB	4s	Follow Packets
151.101.3.6:443	192.168.50.101:60695	SSL TCP TLS	34	12.7 KiB	7.9 KiB	4.8 KiB	2s	Follow Packets
17.156.129.129:443	192.168.50.101:60688	SSL TCP TLS	34	12.7 KiB	9.3 KiB	3.4 KiB	0s	Follow Packets
17.156.129.129:443	192.168.50.101:60697	SSL TCP TLS	31	12.4 KiB	9.2 KiB	3.2 KiB	1s	Follow Packets
192.168.50.101:60691	2.22.144.160:443	SSL TCP TLS	25	8.8 KiB	2.7 KiB	6.1 KiB	30s	Follow Packets
17.167.200.74:443	192.168.50.101:60687	SSL TCP TLS	28	8.8 KiB	5.5 KiB	3.3 KiB	10s	Follow Packets
17.57.163.32:443	192.168.50.101:60690	SSL TCP TLS	21	7.7 KiB	4.8 KiB	2.9 KiB	0s	Follow Packets
17.253.37.201:80	192.168.50.101:60684	HTTP TCP	10	1.4 KiB	733 B	662 B	0s	Follow Packets
17.253.37.201:80	192.168.50.168:52518	HTTP TCP	10	1.4 KiB	733 B	650 B	0s	Follow Packets
192.168.50.101:60685	2.23.210.78:80	HTTP TCP	10	1.2 KiB	715 B	465 B	0s	Follow Packets
17.250.81.26:443	192.168.50.101:60694	TCP	6	416 B	140 B	276 B	0s	Follow Packets
17.250.81.18:443	192.168.50.101:60692	TCP	4	280 B	148 B	132 B	1s	Follow Packets

Figure: Conversations — who spoke to whom, how much each way, sortable, with one-click Follow and filtering.

🔍 The honest truth

“Follow” shows the *transport stream* — the bytes as they crossed the wire. For an HTTPS connection that is **ciphertext**, and PiLens labels it as such. It is honest about the difference between reassembled packets and decrypted content.

DNS Explorer

→ TL;DR

Every cleartext name lookup: who asked, for what, what came back, and whether it failed. Group by name, or filter to errors only.

PiLens network forensics

DNS Explorer
Cleartext DNS observed through the appliance. Encrypted DNS (DoH/DoT) is not shown here; it appears as TLS/HTTPS to the resolver.

Filter name / client / answer ☒ Group by name ☐ Errors only 52 messages

NAME	TYPE	QUERIES	CLIENTS	ANSWERS	RESULT	LAST SEEN
dns.resolver.arpa	SVCB	6	192.168.50.101, 192.168.50.1, 192.168.50.168			10/9/2026, 6:17:31 PM
captive.apple.com	HTTPS	4	192.168.50.101, 192.168.50.1, 192.168.50.168			10/9/2026, 6:17:31 PM
captive.apple.com	A	4	192.168.50.101, 192.168.50.1, 192.168.50.168			10/9/2026, 6:17:31 PM
fp-gb-o2.rcs.telephony.goog	NAPTR	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:20 PM
netcts.cdn-apple.com	HTTPS	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:20 PM
netcts.cdn-apple.com	A	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:20 PM
mask.icloud.com	HTTPS	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:20 PM
mask.icloud.com	A	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:20 PM
mask.apple-dns.net	HTTPS	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:20 PM
ai744.dscw154.akamai.net	HTTPS	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:20 PM
sips.tcp.fp-gb-o2.rcs.telephony.goog	SRV	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:20 PM
fp-gb-o2.rcs.telephony.goog	A	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:20 PM
epdg.epc.mnc010.mcc234.pub.3gppnetwork.org	A	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:20 PM
gs-loc.apple.com	HTTPS	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:23 PM
gs-loc.apple.com	A	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:23 PM
captive.g.aaplimg.com	HTTPS	2	192.168.50.168, 192.168.50.1			10/9/2026, 6:17:31 PM
apple-native-relay.apple.com	HTTPS	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:40 PM
apple-native-relay.apple.com	A	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:40 PM
mask-api.icloud.com	HTTPS	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:41 PM
mask-api.icloud.com	A	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:41 PM
mask-api.fe2.apple-dns.net	A	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:41 PM
apple-native-relay.mask.apple-dns.net	HTTPS	2	192.168.50.101, 192.168.50.1			10/9/2026, 6:17:50 PM

Signed in as admin
v0.3.3 Sign out

Figure: DNS Explorer, grouped by name — queries, answers, record types and result codes.

🔔 Heads-up

If a device uses **encrypted DNS** (DoH/DoT), its lookups are inside TLS and *won't* appear here — they look like HTTPS to a resolver. PiLens never pretends encrypted DNS is visible.

TLS Explorer

→ TL;DR

Every TLS/QUIC handshake: server name (SNI), negotiated version, cipher, ALPN, and any certificate sent in the clear. Metadata only — marked “encrypted pass-through” on every row.

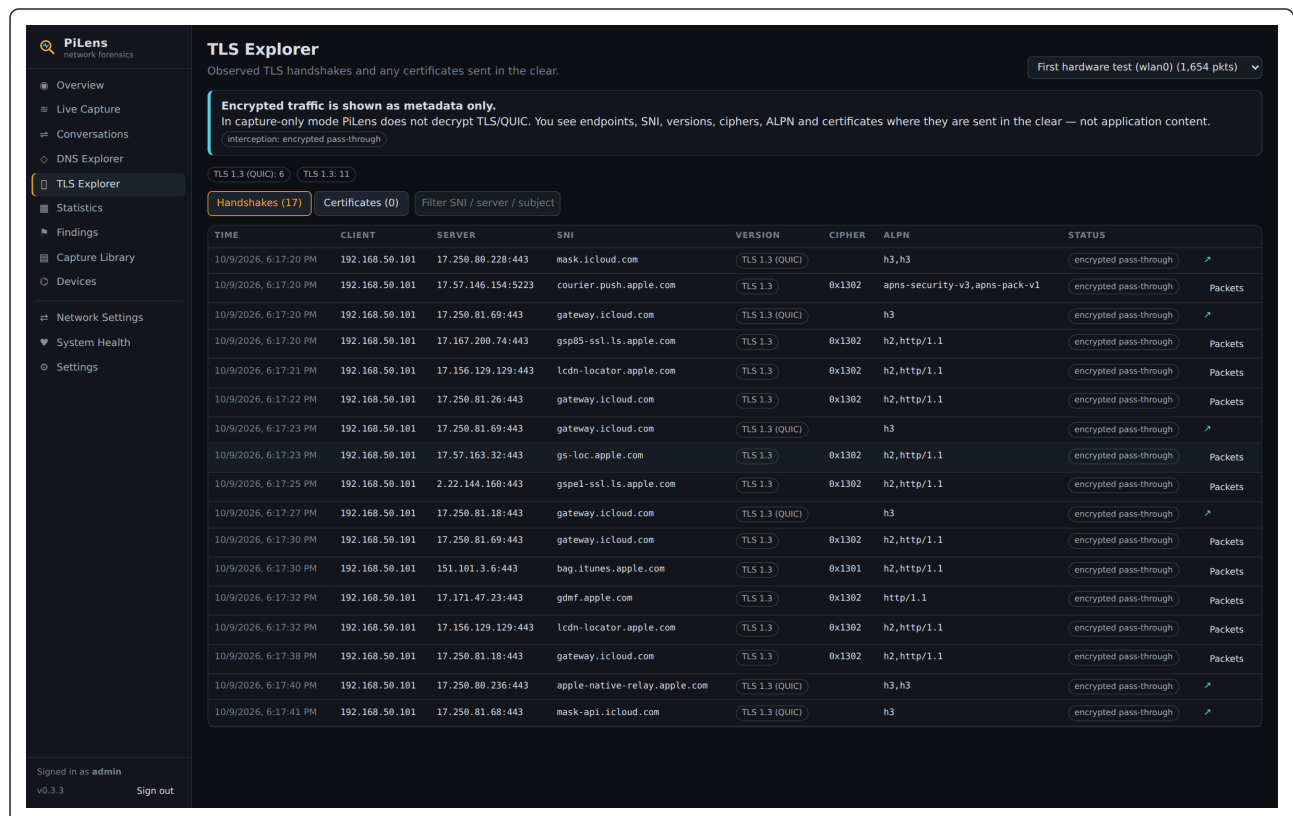


Figure: TLS Explorer. Real server names and versions, honestly labelled. The certificate tab is usually empty on modern servers — TLS 1.3 encrypts the certificate.

Pro tip

The version column reports the *true negotiated* version, not the compatibility value on the wire. QUIC shows as “TLS 1.3 (QUIC)”. If you ever see everything claiming “TLS 1.2”, that’s a bug in a lesser tool — here it’s read from the handshake properly.

Statistics

→ TL;DR

The shape of the capture: traffic over time, protocol mix, top talkers, packet sizes. Toggle bytes vs packets. Every number is a real count, defined in the panel on the right.

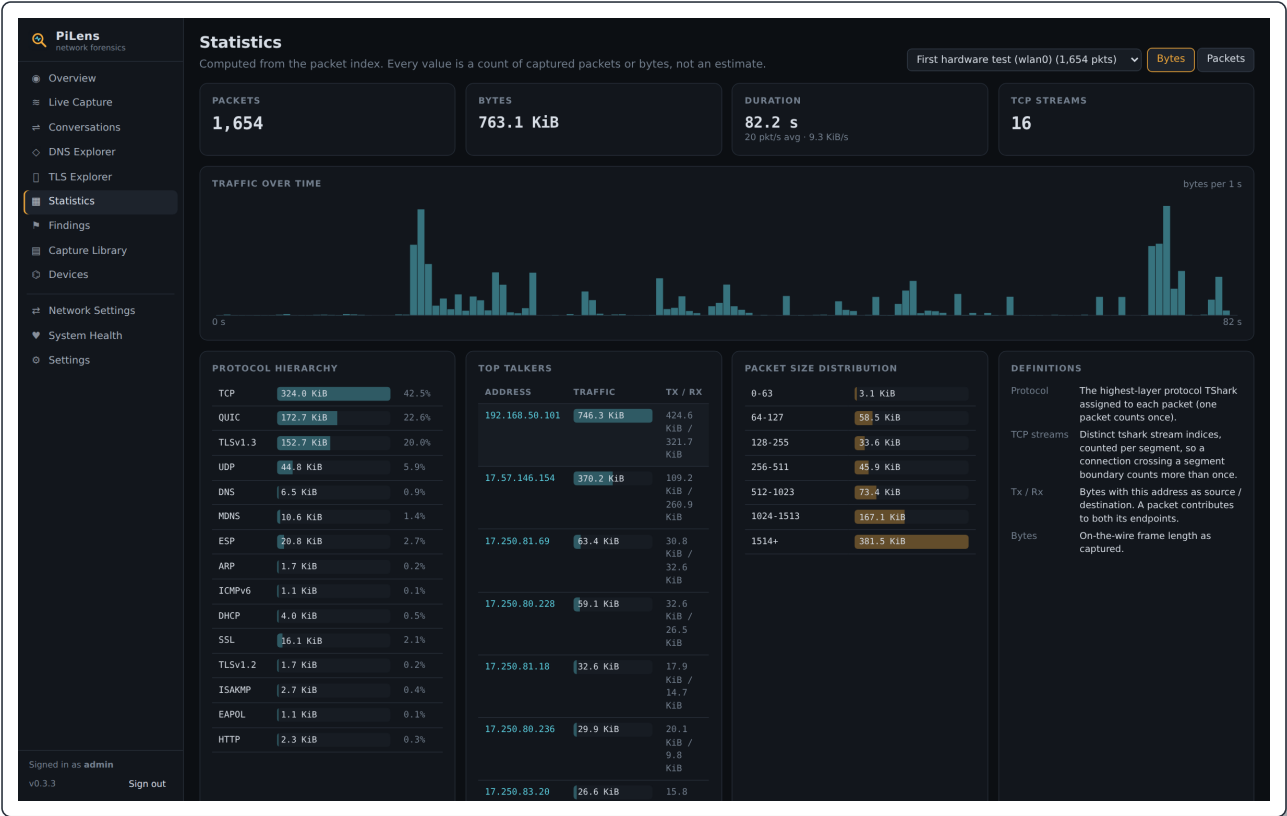


Figure: Statistics — a traffic histogram, protocol hierarchy, top talkers and size distribution, with a definitions panel so nothing is a mystery.

Findings

→ TL;DR

Observable hiccups — retransmissions, resets, SERVFAILs, NXDOMAINs, dropped packets — each with evidence you can click straight to.

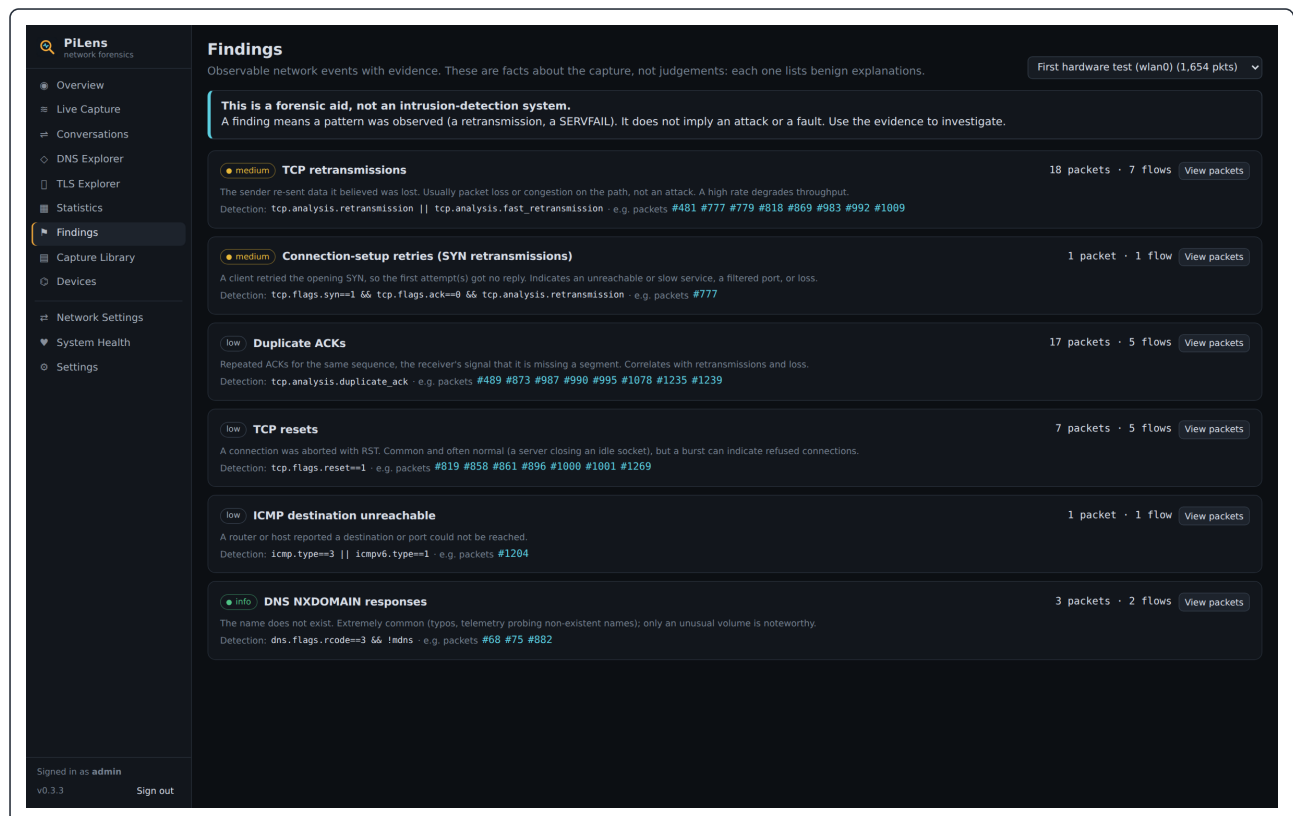


Figure: Findings. Each card is a fact with example packets, a reproducing filter, and a note on benign causes.

The honest truth

Findings is a **magnifying glass, not a smoke alarm**. A finding means a pattern was *observed*, not that something is wrong or malicious. Each card spells out the innocent explanations. PiLens will never tell you “you’ve been hacked” from a retransmission.

HTTPS inspection (Mode B) — the opt-in one

→ TL;DR

A toggle on **Certificates**. Download the appliance’s CA, install and trust it on a device you own, flip the switch, and that device’s HTTPS shows up decrypted in the **HTTP Explorer**. Off by default.

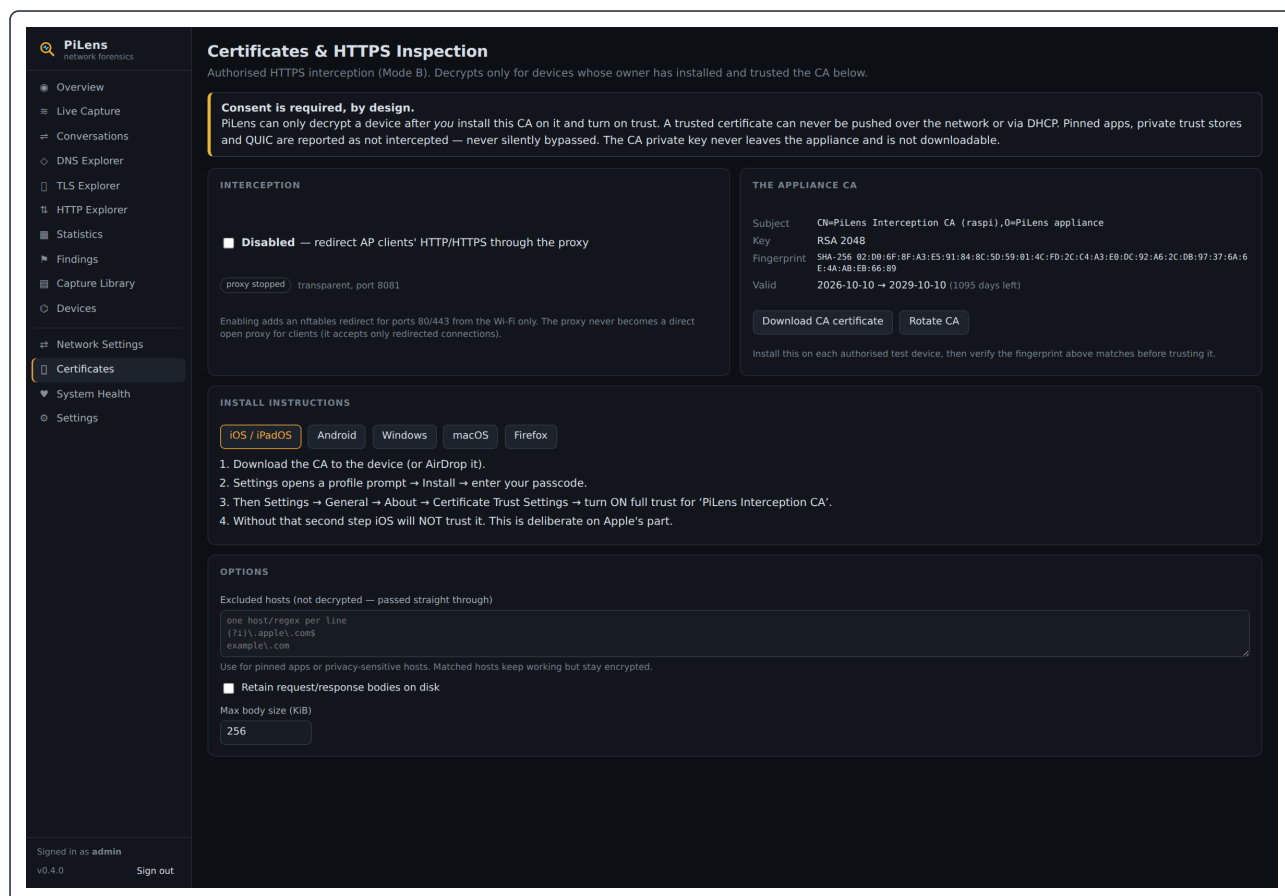


Figure: Certificates & HTTPS Inspection — the toggle, the CA to download, per-device install steps, and the privacy options.

The honest truth

This is the only part that can see inside HTTPS, and it only works *with consent*: you must install the CA on each device by hand and turn trust on. It **cannot** be pushed over the network or via DHCP. Pinned apps, QUIC and private trust stores still can't be read — they show up as honest errors, never silently bypassed. The CA's private key never leaves the Pi and is not downloadable.

To use it: Certificates > *Download CA certificate* > install it on your test device (the page has per-OS steps; iOS needs the extra “Certificate Trust Settings” toggle) > verify the fingerprint matches > flip *Interception* on.

Heads-up

Turning interception on while a device has *not* installed the CA will break that device's HTTPS (it correctly refuses the unknown certificate). Install first, toggle second — and toggle it back off when you're done.

Pro tip

Bodies are off by default. The Explorer shows headers, status and timing for everything; request/response bodies are only stored if you opt in on the Certificates page, for a named investigation. Clear them when done.

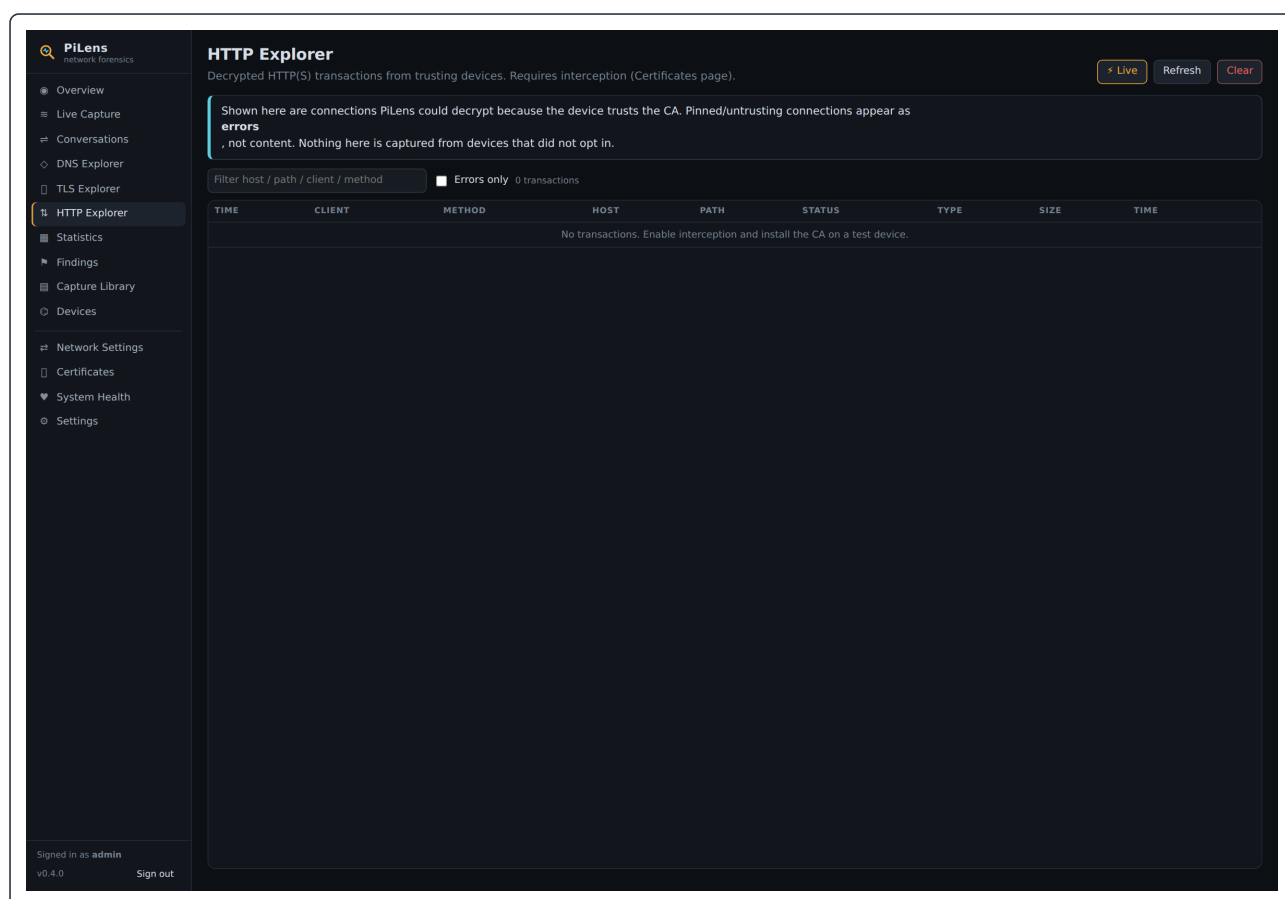


Figure: HTTP Explorer — decrypted transactions from trusting devices, with headers and (opt-in) body viewers.

The housekeeping screens

Capture Library

Every saved session. Rename, tag, add notes, mark *preserved* (never auto-deleted), export, or check integrity (SHA-256 per file + a manifest).

Devices

Who has been on the Wi-Fi. Label them, add notes, jump to their packets.

Certificates

The HTTPS-inspection toggle, CA download and device install guide (§3.9).

Network Settings

Change the access point, DHCP, DNS, forwarding and management access. Applied as one transaction with automatic rollback.

System Health

CPU, memory, temperature, services, storage, interfaces — each with what was actually measured.

Settings

Theme, density, password, two-factor, storage limits, colour rules, and the audit log.

Randomised
MACs show
badge; PiL
never pret
they identif
hardware.

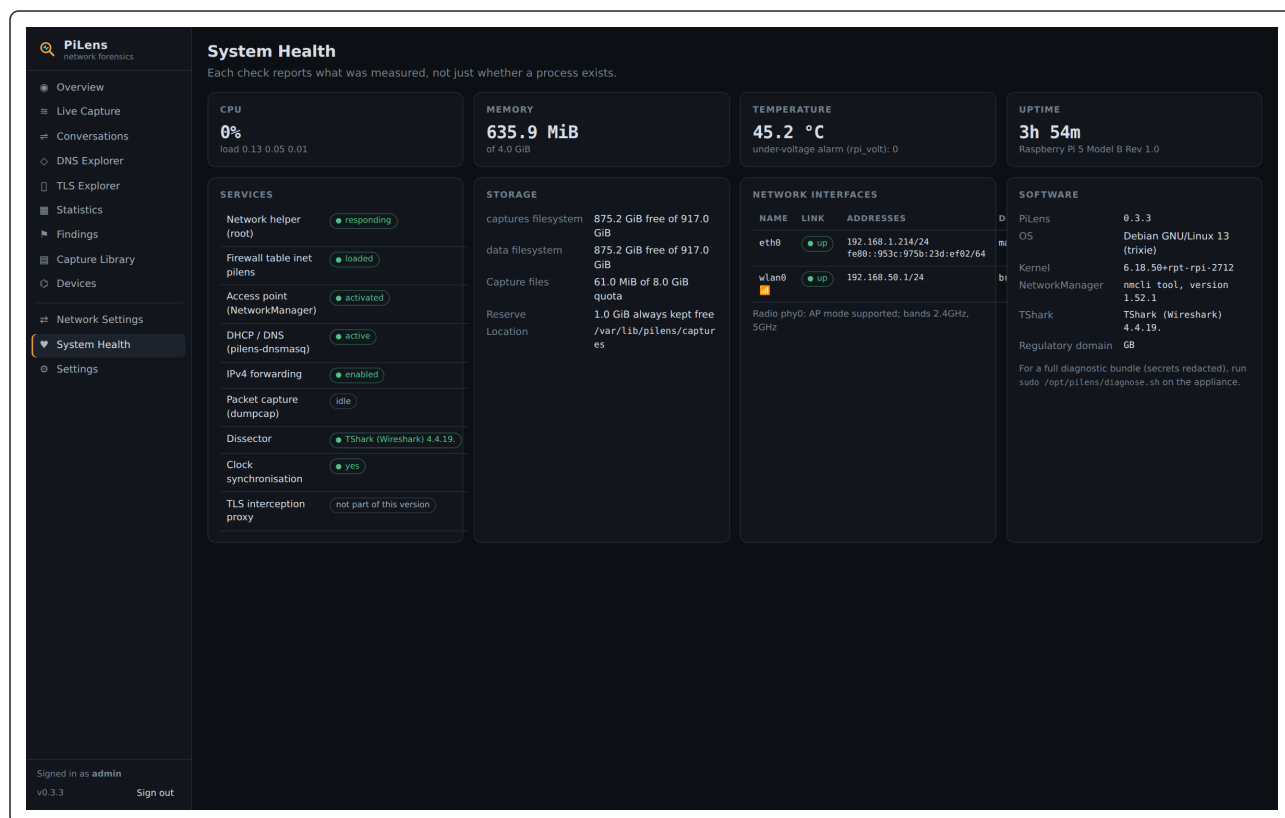


Figure: System Health — every service reports what it actually did, not just whether a process exists.

4 Common tasks, as recipes

→ TL;DR

Short how-tos for the things you'll actually do. Find the one you need, ignore the rest.

“See everything one device does”

1. **Devices** > find it > **Packets**. (Or label it first so you recognise it.)
2. You land in Live Capture filtered to that device. Add `&& dns` or `&& tls` to narrow further.

“Save this capture as evidence”

1. Stop the capture. **Library** > the session > **Edit** > tick **Preserved**, add notes.
2. **Integrity** > **Verify hashes**, then **Download manifest**. The original .pcapng opens in desktop Wireshark too.

“Analyse a capture from somewhere else”

1. **Library** > **Import pcap/pcapng**. It indexes in the background; the file is kept exactly as you uploaded it.

“Stop a device reaching the internet but still watch it”

1. **Network Settings** > **Forwarding** > **No forwarding** > apply. Clients still get an address and DNS, so you can see what they *try* to reach.

5 When something's off

→ TL;DR

Run `sudo /opt/pilens/diagnose.sh` first (it redacts secrets). Then the table.

Symptom	Try this
Browser cert warning	Expected. Match the fingerprint, trust it.
Can't log in	<code>sudo -u pilens /opt/pilens/bin/pilens reset-password -user admin</code>
Locked out by the subnet	On the console: <code>sudo nft delete table inet pilens</code> , fix it in the UI, re-apply.
No internet for clients	Check the Overview Forwarding card and your Ethernet uplink.
AP won't start	Try WPA2 / 2.4 GHz / a legal channel. Failed applies roll back by themselves.
Capture shows drops	Use a capture filter; record to an SSD, not the SD card.
Everything's wonky	<code>sudo /opt/pilens/repair.sh</code> fixes permissions and restarts services.

 Heads-up

If a network change ever strands you, every apply keeps a backup and rolls back on failure, and the last-good firewall is restored at boot before the Wi-Fi comes up. You are rarely more than a reboot from a working appliance.

6 Keyboard cheat sheet

<code>/</code>	focus the filter	<code>Space</code>	pause / resume the view
<code>↑ ↓</code>	move through packets	<code>f</code>	follow newest packet
<code>PgUp PgDn</code>	jump a page	<code>Shift+F</code>	focus mode (hide chrome)
<code>← →</code>	collapse / expand tree	<code>Alt+1..7</code>	jump between pages
<code>Home End</code>	first / last packet	right-click	build a filter

 The honest truth

The one rule that matters: only point PiLens at devices and networks you own or are explicitly allowed to inspect. It has no hidden, covert or impersonation features, on purpose. Captured traffic can hold passwords and private messages — treat your captures like the sensitive data they are, and delete what you don't need. Access and exports are logged.