Episode Book

EPISODE #5532

What Your Browser Isn't Telling You

PUBLISHED	24 September 2026 at 16:47 UTC
DURATION	22:49
LISTEN	myweirdprompts.com/episode/browser-traffic-forensics-tools/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.2
BOOK GENERATED	19 October 2026 at 17:37 UTC

SYNOPSIS

If you want to know what your browser is really sending, DevTools is the wrong instrument — it silently drops the initiator data that answers "which script fired this request." This episode walks through the three tap points for browser traffic (packet capture, intercepting proxies, and browser internals), why the HAR format is an abandoned spec that everything still emits, and the fidelity-versus-visibility tradeoff that comes with terminating TLS in the middle. Then we get into what researchers have actually found: a field registry of 3,919 unique fields across six AI chatbots, and structured tests suggesting privacy modes in Claude, ChatGPT, and Grok still transmit session-linked analytics to third parties.

TRANSCRIPT



Corn

Here's the question that stopped me this morning. If you want to know what your browser is actually telling the internet about you, which tool do you reach for?



Herman

And most people would say DevTools. Open the panel, watch the requests scroll by.



Corn

Which is exactly the wrong answer, and that's what Daniel wants to talk about. He wrote in with a whole thing. Last time we tried to look under the hood of ChatGPT, we rebuilt the picture from what OpenAI open-sources and what other people have documented. Herman called a lot of that reverse engineering, which it was.



Herman

Still is.



Corn

Since then we've done local inference with agent harnesses like Claude Code, and proxy layers that sit in the middle of requests and write them down. Now Daniel wants to go one level down. Privacy researchers and advocates want to know exactly what telemetry websites collect that isn't obvious from the user interface. His framing is good: the request leaves your local network wrapped in HTTPS, but locally the content is sitting there in plain text. You can open DevTools and look at it. It's just not the ideal instrument for this specific purpose. So his question is, for anyone who wants to forensically analyse the traffic going out of browsers or other web clients, what tools are actually worth knowing about?



Herman

That's a real question with a real answer, and the answer starts with the fact that DevTools will quietly lie to you.



Corn

Not lie. Omit.



Herman

Omit in ways that change your conclusion. There are three places you can tap traffic, and none of them is a superset of the others. Packet capture, so tcpdump, Wireshark, tshark. Intercepting proxy, so mitmproxy, Burp, Fiddler, Charles. And browser internals, which is DevTools HAR export, the Chrome DevTools Protocol, NetLog. Three different instruments, three different pictures of the same event.



Corn

And the HTTPS thing confuses people. The wire is encrypted. The tap points aren't.



Herman

Right. Every tap point above the TLS layer sees plaintext by definition, because it's sitting where the browser has already decrypted. Packet capture is the odd one out. It sees the TLS handshake, it sees timing, it sees packet sizes, but it can't see the bodies without session keys. That's the whole reason keylog export exists as a feature in these tools.



Corn

So the spine of this episode, and I want to say it once and then move on. The tool you pick determines the question you can answer. "We collected HAR files" does not identify an instrument. That's a line from a reference page called Measure The Web, and it's the sharpest way anyone has put this.



Herman

Let's start with why DevTools fails, because that's the thing Daniel's already half-suspicious of.



Corn

He is. He says it's not the ideal tool and he's right, but I want the concrete version. What does it actually miss?



Herman

The initiator. In a controlled fixture run, twelve HAR entries, zero of them carried an initiator. Eleven CDP events from the same session, all eleven had one, and five of those had a full JavaScript stack frame. If your question is "which script caused this third-party request," a HAR cannot answer it. Not "answers it poorly." Cannot.



Corn

Which is the entire question a privacy researcher is asking.



Herman

It's the entire question. You're looking at a request to some analytics endpoint and you want to know who fired it. Was it the page's own code, was it a tag manager, was it an extension, was it something injected by a third party three hops down the chain? HAR gives you a timestamp and a URL and shrugs.



Corn

So what's the HAR format itself? Because I went looking and the spec situation is funny.



Herman

It's abandoned. The W3C copy of the HAR 1.2 specification literally opens with the words "DO NOT USE." It says the document was never published by the working group and has been abandoned. There is no successor. There's a proposed 1.3 that has been dormant for years.



Corn

A format that everything emits and nothing owns.



Herman

Everything emits it. Chrome emits it, Firefox emits it, Playwright emits it, mitmproxy can emit it, every bug report on the internet asks you to attach one. And the standard is a document that tells you not to use it. Everything useful in a HAR is a vendor extension. Playwright 1.62.1 emits fields like underscore resource type, underscore security details, underscore server port, underscore web socket messages. Chrome emits a different set. So an analysis script written against Playwright's output will run against a Chrome HAR and silently produce empty columns. No error. Just blank.



Corn

Silent empty columns are the worst failure mode in analysis, because you get a result.



Herman

You get a confident result built on nothing.



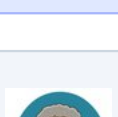
Corn

The sanitization piece is the one that would actually bite a normal person.



Herman

Chrome made sanitized HAR export the default in Chrome 130, October 2024. And it strips exactly three things. Cookie headers, Set-Cookie headers, and Authorization headers. That's it. Query-string tokens survive. POST bodies survive. JWT claims survive. Any non-standard auth header survives.



Corn

So you get a file that feels clean.



Herman

It has a menu option that says sanitized. One wrong click and it re-includes every secret, and even the sanitized version is carrying a lot more than people assume.



Corn

And the size thing, because this is where I actually laughed. Bodies are opt-in, right?



Herman

Opt-in and expensive. Same fixture run: twenty thousand five hundred and sixty-six bytes with bodies omitted, versus two hundred and twenty-six thousand seven hundred and nineteen bytes with bodies embedded. That's eleven times larger for a single two-hundred-kilobyte response. And the mitmproxy HAR in that same run was twelve point eight times the browser's body-less HAR.



Corn

Because the proxy doesn't have the browser's idea of what matters.



Herman

The proxy records everything it saw. The browser records what it decided to tell you about.



Corn

There's a subtler one I want on the record. A HAR is a transaction log, not a cookie jar. If a script sets a cookie via document dot cookie, that cookie appears in no HAR entry, because it never crossed the network. It was created in the page.



Herman

Which means you can read a HAR top to bottom and miss the fact that the page planted an identifier on you.



Corn

And HAR stops at HTTP. Below that there's a whole world.



Herman

Chrome's NetLog goes underneath. DNS stalls, socket-pool waits, QUIC session events, disk-cache reads. You capture it through the net-export page or the log-net-log flag, and you read it in the netlog viewer. That's the layer where you find out the request you thought took two hundred milliseconds actually spent a hundred and eighty of them waiting for a DNS answer that came from a cache you didn't know about.



Corn

So DevTools is a window, not an instrument. Now the proxy tier, because that's where the field actually lives.



Herman

mitmproxy is the default. Ninety-eight papers in a corpus of five thousand eight hundred and fifty-nine papers across CCS, IMC, NDSS, PoPETs, USENIX Security, TheWebConf, and IEEE S&P, from 2010 through this year. Burp Suite appears in eighteen. Fiddler in fifteen. Charles in five.



Corn

Ninety-eight to eighteen. That's not a preference, that's a consensus.



Herman

It's the instrument the field standardized on, partly because it's scriptable and partly because it's been maintained continuously. Version 12.2.3 shipped in May. Three tools in the box: mitmproxy is the console, mitmweb is the graphical one, and mitmdump is the non-interactive one, which people describe as tcpdump for HTTP. It writes a flows stream, and it'll write HAR too if you set the hardump option.



Corn

Now the paradox, because this is the part I want Daniel to hear.



Herman

The proxy terminates TLS in the middle. That's how it sees plaintext bodies by default. But terminating TLS means it is no longer the browser. It presents a new TLS fingerprint. It presents a certificate signed by a custom certificate authority you installed. It may force a QUIC downgrade because the proxy doesn't speak QUIC the way Chrome does.



Corn

So the site can tell.



Herman

The site can absolutely tell. And if your research question is "what do sites do to a normal browser," an intercepting proxy is the one instrument that can invalidate the answer. You've changed the subject of your own study.



Corn

Fidelity versus visibility, and you can't have both.



Herman

You pick which one you're willing to lose. That's the tradeoff the whole episode runs on.



Corn

There's a tier above mitmproxy, though, which is the research-grade stuff. OpenWPM.



Herman

OpenWPM is the closest thing web privacy measurement has to a standard instrument. It's a Python platform that drives an unbranded Firefox through Selenium and records through a privileged WebExtension. It powered the one-million-site tracking study, Englehardt and Narayanan, CCS 2016, which is the paper that basically founded the modern version of this field.



Corn

Unbranded Firefox is the detail I like. They strip the branding so the sites don't recognize the automation.



Herman

And there's webXray from Timothy Libert, which has been used for hundreds of studies, and tracker-lens, which is a headless Chromium audit using a fingerprint database derived from EasyPrivacy and Disconnect dot me.



Corn

So that's the instrument problem. Three tap points, one abandoned spec, and a proxy that changes what it measures.



Herman

Now the fun part. What people have actually found with these instruments.



Corn

The one I keep coming back to is a repository called har-forensics, created in March of this year. It documents what AI chatbots transmit beyond the visible conversation.



Herman

System prompts, experiment assignments, rate-limit configurations, model routing decisions, and analytics events. Across Claude, ChatGPT, Gemini, Grok, DeepSeek, and Perplexity. And there's a field classification registry in there with three thousand nine



Herman (continued)

hundred and nineteen unique fields.



Corn

Three thousand nine hundred and nineteen fields, for six chat products.



Herman

And the vendors are identifiable. Claude routes analytics through Segment via a subdomain of anthropic dot com. ChatGPT routes through Segment via a path under chatgpt dot com. Grok uses Sentry plus an events endpoint.



Corn

So you can see the plumbing.



Herman

You can see the plumbing. And here's the finding that matters. On the fourteenth of March this year, the same project ran structured tests across Claude's Incognito Chat, ChatGPT's Temporary Chat, and Grok's Private Chat.



Corn

The privacy modes.



Herman

All three. And privacy-mode use was still associated with session-linked analytics transmissions to third parties that are not named in the privacy-mode disclosures. No observable reduction in account-identity fields, no observable reduction in session-linked fields. And in two of the three platforms, activating the privacy feature was itself immediately followed by a third-party analytics transmission.



Corn

So the act of hiding was logged.



Herman

The act of hiding was the event.



Corn

Now I want to be careful here, because this is one researcher's captures.



Herman

It is. The repository has zero stars and one fork. It has not been independently replicated. And the repo itself is scrupulous about this. Its own language is that the captures show transmission, and they do not establish downstream use, retention, intent, or legal violation.



Corn

Which is the right caveat and also doesn't make the transmission not happen.



Herman

The transmission happened. What it means downstream is a separate question and nobody has answered it.



Corn

The repo's framing of the issue is good, actually. It says the issue isn't only what the platform says about training or history, it's whether the consumer-facing disclosure meaningfully captures the broader session architecture that still operates when a user deliberately activates a privacy-protective feature.



Herman

That's the sentence. That's the whole finding in one line.



Corn

Let's go to the agent harnesses, because that's where this gets personal for our audience.



Herman

There's a write-up from August, someone who put GitHub Copilot behind a mitmproxy and reconciled what he saw against the VS Code source. His line is worth quoting: source code tells you what an app can do; discovering what it actually does at runtime is more challenging.



Corn

That's the entire justification for this episode.



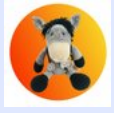
Herman

And what he found at runtime is that recent edits can pull context from files other than the one being edited. Including, in his words, the infamous dot env.



Corn

So you edit one file and the agent ships your environment variables.



Herman

Depending on the configuration, yes. He also located the SQLite session store sitting behind the Chronicle component, which is where the conversation state actually lives on disk.



Corn

There's a counterpoint in that same discussion that I think is fair. Someone pointed out that Gemini CLI is open source, so you don't need to intercept at the network at all. You can just add an inspection call in the source and watch the request object directly.



Herman

That's a legitimate point and it's the right approach when the client is open. But it only works when the client is open. Copilot's extension is not, and neither is the server side of any of these. And even with open source, you're reading what the code says it will send, not what the deployed build actually sends. Those can diverge.



Corn

The tooling ecosystem around this exploded this year.



Herman

It did. There's a project called ccproxy, three hundred and forty-seven stars, which is a transparent interceptor built on mitmproxy plus WireGuard with full TLS inspection and Wireshark keylog export. It jails a process in a rootless WireGuard namespace and intercepts at the network layer, so the process doesn't know it's being watched.



Corn

Rootless is the important word.



Herman

Rootless means you don't need to reconfigure the whole machine. And it has two features that are clever. Flows diff, which does a sliding-window diff of request bodies so you can see what changed between two otherwise identical requests. And flows compare, which diffs the request the client sent against the request that got forwarded.



Corn

Which is how you catch a middle layer rewriting things.



Herman

That's exactly what it's for.



Corn

Then there's Sherlock, which hit the front page in January. Sits between the LLM tools and the API, live dashboard, auto-saves every prompt as markdown or JSON.



Herman

And that one has a problem. A commenter flagged that it unconditionally disables TLS verification for upstream requests. His words were that you should not use it.



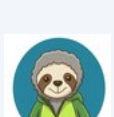
Corn

And that's the meta-point of the whole segment. The tools you use to audit privacy are themselves mostly unaudited.



Herman

A tool that turns off certificate verification to make its own life easier has just removed the one protection standing between the user and whoever is on the path. It's an easily accessible front door.



Corn

There's a better-behaved example. LocalAI ships a MITM proxy that redacts personally identifiable information from Claude Code and Codex traffic without ever holding the API keys. It intercepts only allowlisted LLM endpoints, api dot anthropic dot com and api dot openai dot com by default, and passes everything else through as a plain TCP tunnel.



Herman

That's the design I'd point people at. Narrow interception, explicit allowlist, no key custody, everything else untouched.



Corn

Now the second-order risk, which is the part I think Daniel is circling without quite naming. What happens when you collect this data yourself?



Herman

You've built a credential store. A HAR file is a live secret. And there's a case that proves it. Okta, between the twenty-eighth of September and the seventeenth of October 2023. An attacker used session tokens pulled out of customer-uploaded HAR files to hijack five customers' sessions.



Corn

Customer-uploaded. As in, support asked for a HAR and the customer sent one.



Herman

That's the vector. The initial disclosure said a hundred and thirty-four customers' files had been accessed. By the twenty-ninth of November that had been revised to essentially all Workforce and Customer Identity customers.



Corn

Which is a support workflow problem dressed up as a security incident.



Herman

It's a support workflow problem. Someone wrote "please attach a HAR file" into a template, and that template turned every support ticket into a credential handoff.



Corn

And if you want to clean one before sharing it, good luck.



Herman

There is no maintained general-purpose HAR sanitizer as of the middle of August. Google's and Cloudflare's har-sanitizer, sanitizhar, Beyond Identity's har-sanitize, harmor, Edgio's har-tools. All archived or dormant. Every one of them.



Corn

So the artifact everyone is asked to produce is a credential store with no maintained tool for cleaning it.



Herman

That's the state of the world.



Corn

There's one more measurement problem, and it's the one that undermines a lot of published work.



Herman

Replay fidelity. A study from last year, Hantke and colleagues at USENIX Security, looked at how faithfully a recorded session can be replayed. HAR-based replay showed thirteen point nine percent average per-origin difference. Direct execution recording showed zero point seven percent.



Corn

Thirteen point nine versus zero point seven. That's a twenty-fold gap.



Herman

So if you're replaying a page from a HAR and drawing conclusions, you're drawing them from a session that is meaningfully not the session that happened.



Corn

And the gap nobody has filled: there's no quantified comparison of browser HAR versus proxy HAR disagreement at scale on real sites. The Measure The Web reference page explicitly flags it as wanted.



Herman

Wanted, as in nobody has done it. Two instruments, both in wide use, and no published number on how far apart they are.



Corn

That's a strange hole for a field this mature.



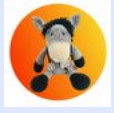
Herman

It's a very strange hole. And it's the kind of thing that only looks obvious once someone does it.



Corn

So the instrument problem and the findings problem are the same problem. If you don't know which tap point produced your data, you don't know what your data can support.



Herman

Most published work doesn't say. Twelve point seven percent of capture-tool uses in that corpus state a version number at all. For tcpdump it's three point four percent. For mitmproxy it's twenty-four point eight.



Corn

Three-quarters of mitmproxy papers don't say which mitmproxy.



Herman

Three-quarters. In a field where the tool's behavior has changed across versions.



Corn

That's where the wire becomes the only honest witness.



Hilbert

Except it isn't a witness. It's a deponent.



Corn

...Go on.



Hilbert

I did a stint doing e-discovery support for a mid-size litigation firm. My job was pulling HAR files and proxy captures off client machines and turning them into exhibits. I still call a HAR file a deposition you didn't know you were giving.



Herman

That's a better framing than anything we've said.



Hilbert

The problem isn't the spec and it isn't the sanitization. Those are real, but they're not the problem. The problem is that a HAR file is a document. Documents get subpoenaed. I watched a paralegal open one in a text editor and scroll right past a session cookie without recognizing it. That's the whole Okta thing in one image.



Corn

The refresh token.



Hilbert

We had a client whose HAR file had a full OAuth refresh token sitting in a query string. Plain as day. The firm's response was to stop collecting HAR files entirely and go back to screenshots.



Herman

Which is the wrong lesson.



Hilbert

It's the wrong lesson and the right instinct. You lose the initiator, you lose the timing, you lose everything that made it useful. But nobody was going to train forty paralegals to read a HAR file, and they were right about that.



Corn

The artifact is a liability the moment it exists.



Hilbert

The artifact is a liability the moment it exists. Someone is waiting for me in the car park, so.



Corn

Right. So the gap Daniel's question exposes is that we have three instruments, no shared standard, and no maintained way to clean the output of any of them.



Herman

The field is moving toward the wire precisely because the clients are getting less inspectable, not more. As agent harnesses pull context from more files, the only place that sees all of it is the network.



Corn

Which is why the tools for reading it matter more every month, and why it's worth noticing that most of them are one person's side project.



Herman

ccproxy is one repository. har-forensics is one researcher. Sherlock is one developer, and it shipped with certificate verification off.



Corn

The instruments are load-bearing and the instruments are hobbyist.



Herman

That's the honest summary.



Corn

Thanks to Hilbert Flumingtop, who produces this show and occasionally deposes it. This has been My Weird Prompts.



Herman

If you want to argue with us about tap points, email us at show at my weird prompts dot com. We'll be back soon.