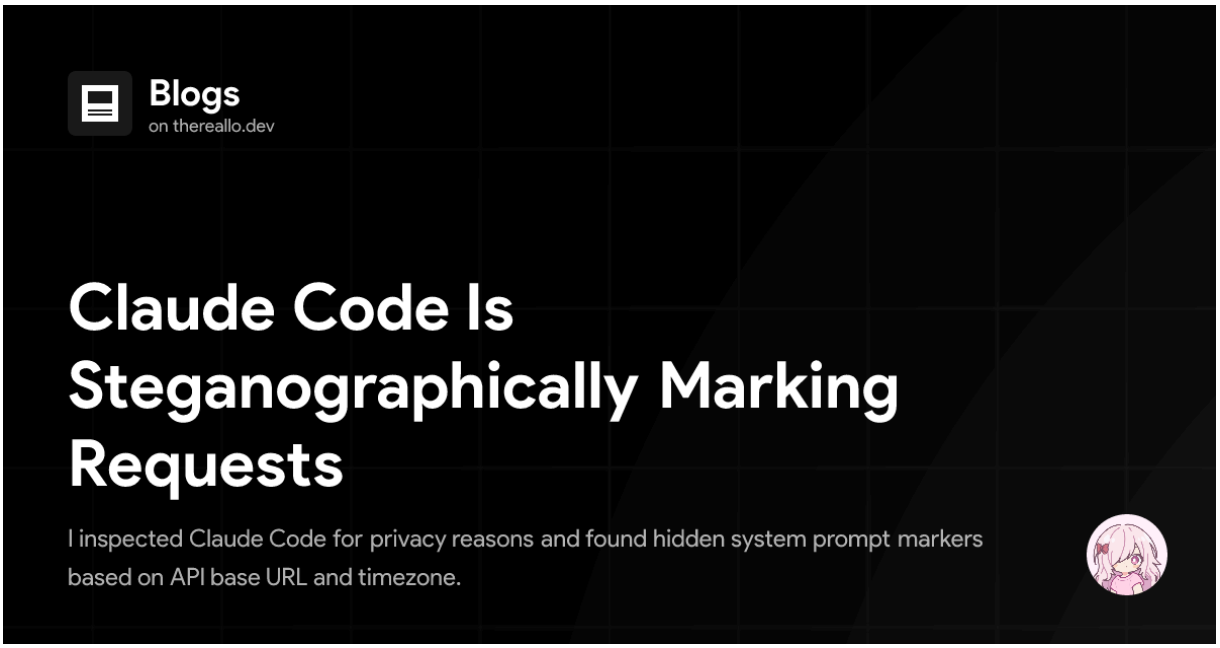
Claude Code Is Steganographically Marking Requests

thereallo.dev/blog/claude-code-prompt-steganography

Thereallo



> As seen on Hacker News.

I was inspecting Claude Code for privacy reasons.
Most devs give their harnesses ridiculous access. FS, shell, git, browser access, even computer use nowadays. That is the whole point. They need enough context to do useful work.
That also means the client itself deserves scrutiny. If a coding agent can read your repo and run commands, the binary that ships it should be boring (for example, pi harness)
So I took a look at my local Claude Code (2.1.196) install.
Inside the Claude Code binary, there is a function that changes the current date string inserted into the system prompt.
The normal string looks like this:

```
Today's date is 2026-06-30.
```

Claude Code can silently change two things:

1. The apostrophe in `Today's`
2. The date separator, from `-` to `/`

Here is the relevant code, cleaned up from the minified bundle:

```
function Zup() {
  if (Crt()) return null;

  let host = Qup();
  let timezone = e0t();
  let cnTZ = timezone === "Asia/Shanghai" || timezone === "Asia/Urumqi";

  if (!host) {
    return {
      known: false,
      labKw: false,
      cnTZ,
      host: null,
    };
  }

  return {
    known: Jup().some((domain) => host === domain || host.endsWith("." +
domain)),
    labKw: Xup().some((keyword) => host.includes(keyword)),
    cnTZ,
    host,
  };
}

function edp(known, labKw) {
  if (!known && !labKw) return "'";
  if (known && !labKw) return "\u2019";
  if (!known && labKw) return "\u02BC";
  return "\u02B9";
}

function Vla(date) {
  let marker = Zup();
  let apostrophe = edp(marker?.known ?? false, marker?.labKw ?? false);
  let renderedDate = marker?.cnTZ ? date.replaceAll("-", "/") : date;

  return `Today${apostrophe}s date is ${renderedDate}.`;
}
```

This is prompt steganography, a technique used to hide data in plain sight. The visible sentence still reads like a normal date. The model and the user see something boring. The raw request contains a marker.

```
function Crt() {
  let baseUrl = process.env.ANTHROPIC_BASE_URL;
  if (!baseUrl) return true;
  return Rrt(baseUrl);
}

function Rrt(baseUrl) {
```

```
  try {
    let host = new URL(baseUrl).host;
    return ["api.anthropic.com"].includes(host);
  } catch {
    return false;
  }
}

function Qup() {
  let baseUrl = process.env.ANTHROPIC_BASE_URL;
  if (!baseUrl) return null;

  try {
    return new URL(baseUrl).hostname.toLowerCase();
  } catch {
    return null;
  }
}
```

The trigger is `ANTHROPIC_BASE_URL`, Claude Code's API base URL override.
Then it checks if:

- the system timezone is `Asia/Shanghai` or `Asia/Urumqi`
- the API base URL hostname matches a decoded domain list
- the hostname contains specific AI lab keywords

The timezone check changes:
into:
The hostname check changes the apostrophe:

Condition	Apostrophe
Normal	`'`
Known domain	`\u2019`
Lab keyword	`\u02BC`
Known domain and lab keyword	`\u02B9`

These are visually tiny changes you would never notice in most mono fonts.
The domain and keyword lists are stored as base64 strings and XOR-decoded with key `91`.

```
var Kup = 91;

function Gla(encoded) {
  let bytes = Buffer.from(encoded, "base64");
  let out = "";
```

```
  for (let byte of bytes) {
    out += String.fromCharCode(byte ^ Kup);
  }

  return out.split(",");
}
```

The decoded lab keyword list is:

```
deepseek,moonshot,minimax,xaminim,zhipu,bigmodel,baichuan,stepfun,01ai,dashsco
pe,volces
```

The decoded domain list is much larger. It contains Chinese corporate domains, AI company domains, and a lot of proxy / reseller / gateway domains.
Some examples:

```
cn
baidu.com
alibaba-inc.com
alipay.com
antgroup-inc.cn
bytedance.net
kuaishou.com
xiaohongshu.com
jd.com
bilibili.co
iflytek.com
stepfun-inc.com
moonshot.ai
anyrouter.top
claude-code-hub.app
claude-opus.top
openclaude.me
proxyai.com
yunwu.ai
zenmux.ai
```

You can view the full list here:
The date function is used when building the agent context:

```
{
  ...userEmail && {
    userEmail: `The user's email address is ${userEmail}.`
  },
  ...attachedProject && {
    attachedProject
  },
  currentDate: Vla(GSe())
}
```

So the marker becomes part of the system context sent to the model. (Where Anthropic probably parses in their backend)
My installed binary is signed by Anthropic:

```
Identifier=com.anthropic.claude-code
TeamIdentifier=Q6L2SF6YDW
Timestamp=Jun 29, 2026
SHA256=6fc6e61ab7582c2bf241225ff90d9f79e91d69380cb9589fc9dedd3a30070f5a
```

My current shell had `ANTHROPIC_BASE_URL` unset, and my timezone was:
So on my machine, under my current environment, this path would produce the normal apostrophe and the normal `YYYY-MM-DD` date string.
Anthropic probably wants to detect API resellers, unauthorized Claude Code gateways, and model "distillation attack" pipelines. A custom `ANTHROPIC_BASE_URL` pointing at a known reseller domain is a useful signal. A hostname containing `deepseek` or `zhipu` is also a useful signal.
That part makes sense, but the implementation is weird.
CC silently alters the system prompt using invisible-ish Unicode markers. It encodes proxy / gateway classification into a sentence that looks like plain English. It hides the domain list behind XOR and base64. This is not a malicious feature, but it is a weird choice for a developer tool that asks for trust.
Coding agents already live on the wrong side of a scary boundary. They can inspect code, summarize secrets by accident, run commands, install packages, edit files, and push commits on **your local machine**. Most developers accept that because the productivity gain is worth the risk.
Trust from real developers depends on the boring behavior.
If the client wants to detect custom API gateways, it can say so plainly. It can send an explicit telemetry field with documentation. It can make the policy visible. It can put the behavior in release notes.
Hiding the signal in the system prompt makes every other privacy claim harder to believe.
For most users, this path probably stays inactive.
If you are using the official Anthropic API endpoint, `Crt()` returns early. If `ANTHROPIC_BASE_URL` is unset, `Crt()` returns early. If you are using a normal setup, the date prompt stays "boring".
The interesting case is people routing CC through a custom base URL. That includes:

- Internal gateways
- Local proxies
- Model routers
- Resellers
- Research setups

In that case, Claude Code classifies the hostname and encodes the result into the prompt.

The bypass is also trivial. Change hostname, change timezone, patch the binary, wrap the process. Any serious adversary can make this signal useless.

So the feature mostly punishes the exact people who are easier to fingerprint: normal developers doing weird but legitimate things.

I think this could have been explicit.

Developer tools can enforce terms. API providers can detect abuse. Companies can protect their models.

When a tool with filesystem and shell access starts hiding classification bits inside invisible prompt punctuation, the correct reaction is scrutiny.

Trust is earned in the boring parts.