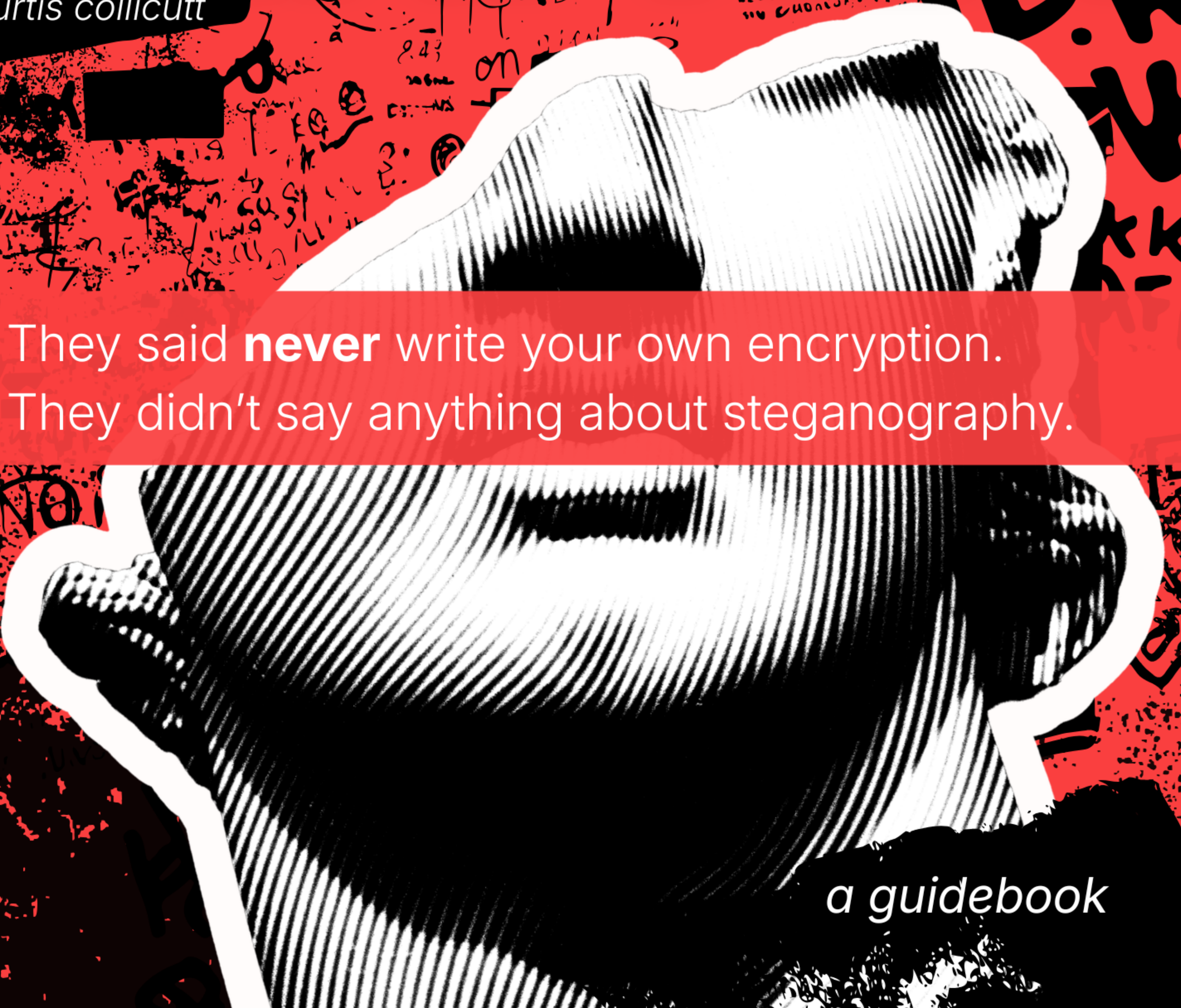
watermarks & auxilary signals in log files

collicutt.net

STEGALOG
GRAPHER

curtis collicutt



They said **never** write your own encryption.
They didn't say anything about steganography.

a guidebook

Stegalographer

They said never write your own encryption.
They didn’t say anything about steganography.

By Curtis Collicutt

Table of Contents

Introduction to this Guidebook

Overview

Steganography is the art of hiding in plain sight. Unlike encryption, which scrambles a message into ciphertext, steganography conceals the very existence of communication. Stegalographer, a specific software project, is an application of steganography, and it applies this principle to one of the most ubiquitous and overlooked artifacts in computing: log files. Every server, every application, every system generates logs. They scroll past endlessly, typically examined only when something breaks. They are infrastructure noise, like background radiation, the exhaust of running software. And that makes them the perfect hiding place.

Stegalographer places a difficult to discover auxiliary signal, a message or messages, inside legitimate-looking log entries. The output is indistinguishable from real application logs because it is real log structure. Your data is encoded not in the content of the logs, but in the choices: which log patterns appear, how timestamps vary by microseconds, which synonyms are selected, where whitespace falls. The message dissolves into decisions that no one thinks to question.

Importantly, steganography has multiple applications, including not only the encoding of auxiliary signals, but also concepts such as digital watermarking. These applications are wide-ranging, so we should consider not only the placement of messages, but also tampering protections, watermarking and other steganographic capabilities, and the Stegalographer project can also be used in these situations.

This guidebook documents the Stegalographer project in full. It explains not just how to use the tool, but how it works: the algorithms that parse log structure, the methods that encode bits into formatting choices, the tradeoffs between capacity and stealth. Whether you need to understand Stegalographer for authorized security testing, covert communication research, or simply to satisfy your curiosity about what is possible when you look at familiar things from an adversarial angle, this guide will take you there.

What is a Guidebook?

This document, a “guidebook”, is the artifact of a learning journey. It documents the process of exploring steganography, covering its history, the underlying computer science and philosophy, and the practical challenges involved in developing a steganographic application such as Stegalographer. It is not an in-depth or rigorous academic study as it is not peer-reviewed and will almost certainly contain mistakes. Consider it a companion to the Stegalographer project itself, perhaps just a big README, as well as possibly a starting point for your own reading and research.

Rather than being exhaustive, it is intended to provide a guided overview of the application and its computing and computer science foundations, notably from an amateur perspective. To some extent, it is based on the idea that “you can just do things”, and that one way to learn about something is to build it and while doing so explore its fundamental makeup.

Caveats and Notices

- *Stegalographer is an imperfect tool and, by its very nature, can never be perfect. In its current form, it is an experiment. It should not be used in real-world environments without further testing to validate and improve its robustness.*

- *This guidebook is a living document. I have done my best to footnote all the external content I have used, but ultimately this should be considered as a more in-depth project README, and not a peer-reviewed paper.*