# HPACK Security Observations

Eric Rescorla

Mozilla

ekr@rtfm.com

## Threat Model Overview (Abstract)

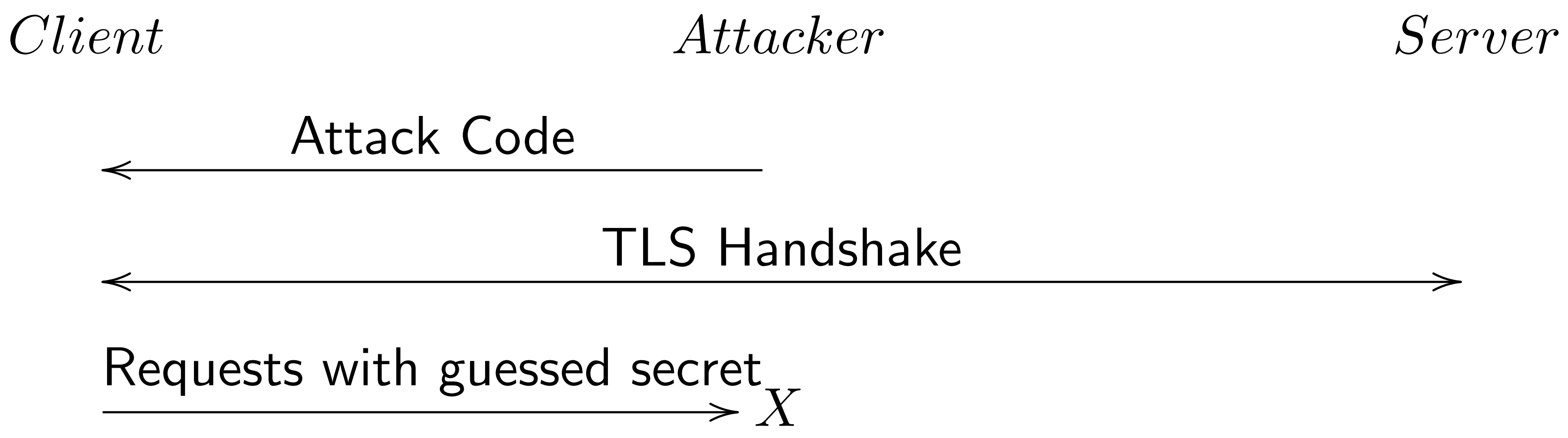
- Compression function acts as an oracle
- Attacker can *query* the oracle to see if a given header/value is present
- Gets one bit in response: *was this value seen before?*

- What can the attacker do?
  - Search the space of a given header
  - Only works with low entropy secrets

## Is there any way to exploit this in HTTPS?

- Cookies are pretty high entropy
- But what about passwords?
  - Often very low entropy
  - Appear in headers with basic auth (how commonly is this used?)
  - Absent CORS JS probably can't modify this header
- What about other headers?
- And other technologies? (Flash, etc.)

## Why is this stronger than just querying the server?



- Most servers have mechanisms to prevent fast guessing attacks
  - Rate limiting, limited try, etc.
- The attacker allows the client and server to set up an HTTP2/TLS connection
- The attacker injects queries with their guesses
  - But blocks the client's requests to the server
  - So the server never sees the guesses

## Some words from Adam Barth

"The situation gets worse if we consider non-standard web technology, such as Flash. For example, Flash's URLRequest API lets the attacker set a wide variety of headers because it uses a header blacklist rather than a whitelist [2]. Worse, Flash permits the attacker to issue such requests across origins via the navigateToURL API. It just so happens that the Authorization header is on Flash's header blacklist, but we need to consider the possibility that web sites will store sensitive information in headers that aren't on Flash's blacklist.

One reaction I can imagine to this issue is to blame Flash and decry its use of a blacklist rather than a whitelist for security, but that misses the larger point that HPACK weakens security because it requires all downstream technologies to maintain more invariants in order to avoid leaking sensitive information out of an otherwise secure channel."