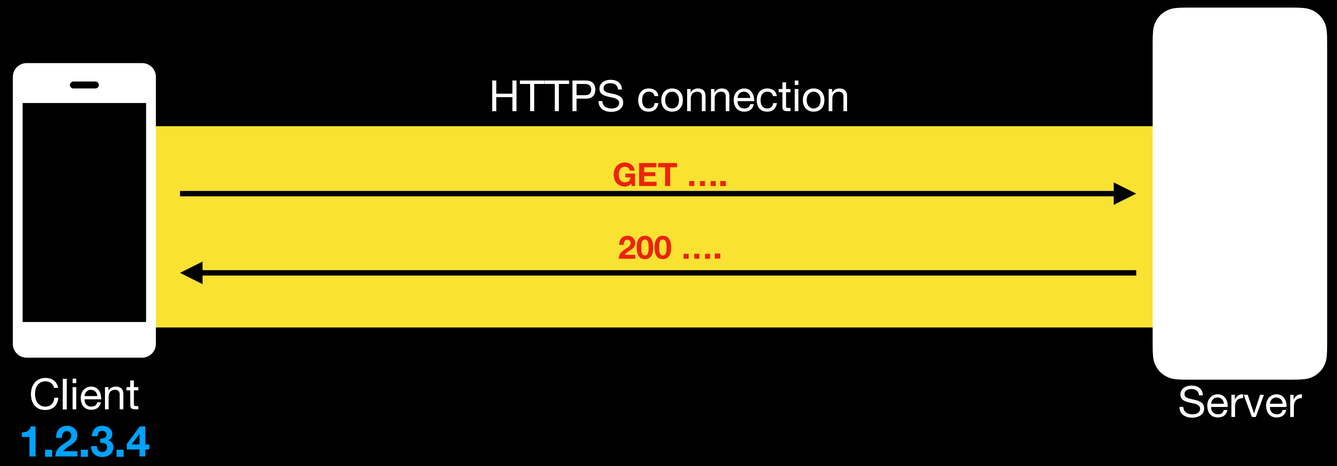
Oblivious HTTP

draft-thomson-ohai-ohttp

Thomson, Wood — IETF 112 Online — OHAI

Problem Statement

Background



Linking client ***data*** with client ***identity*** is problematic for privacy

- DNS requests or Safe Browsing queries reveal browsing history
- Telemetry data reveals client-specific information

Many applications use HTTP for performing *transactional* tasks

Existing Technologies
Background

General-purpose connection-oriented proxies (CONNECT, SOCKS, Tor)

- Often includes stronger requirements and adds more overhead
- Forces a trade-off between connection setup overhead and linkability caused by long-lived connections

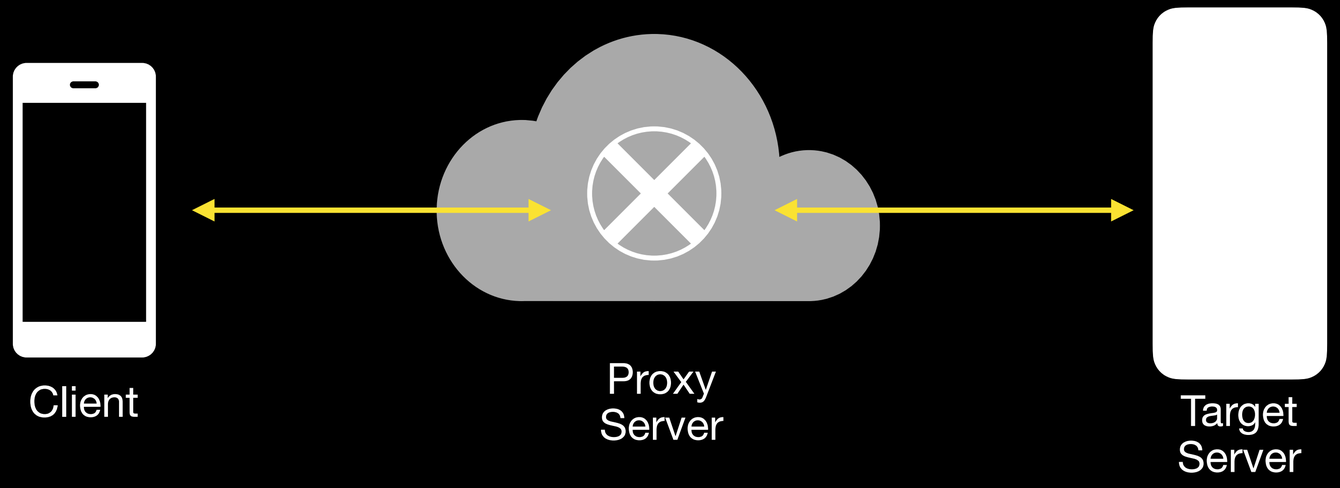
Application-specific protocols (private telemetry collection)

- Adds delay and requires non-trivial infrastructure

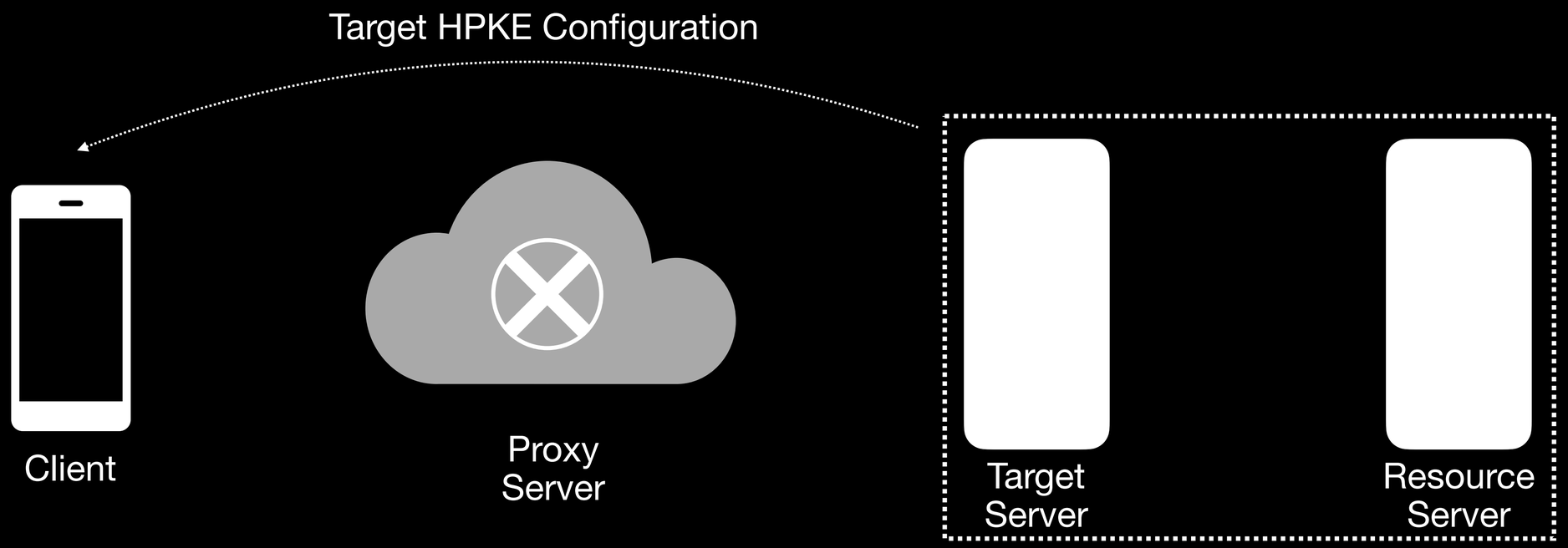
Oblivious HTTP
Protocol

Message-oriented HTTP proxy protocol for *transactional* applications

Splits knowledge of client identity and client data using a *network proxy* and *public key encryption*

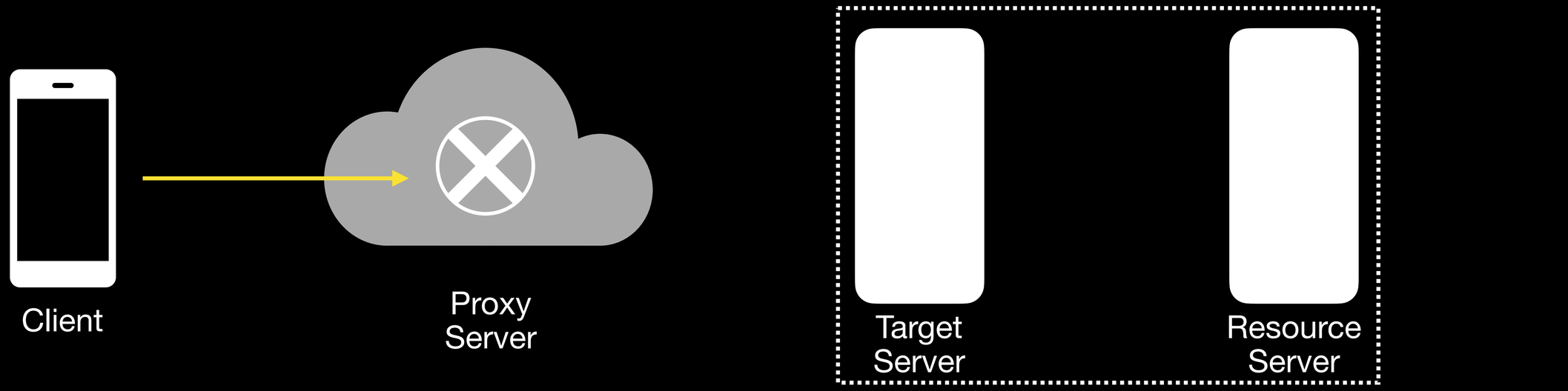


Flow
Protocol



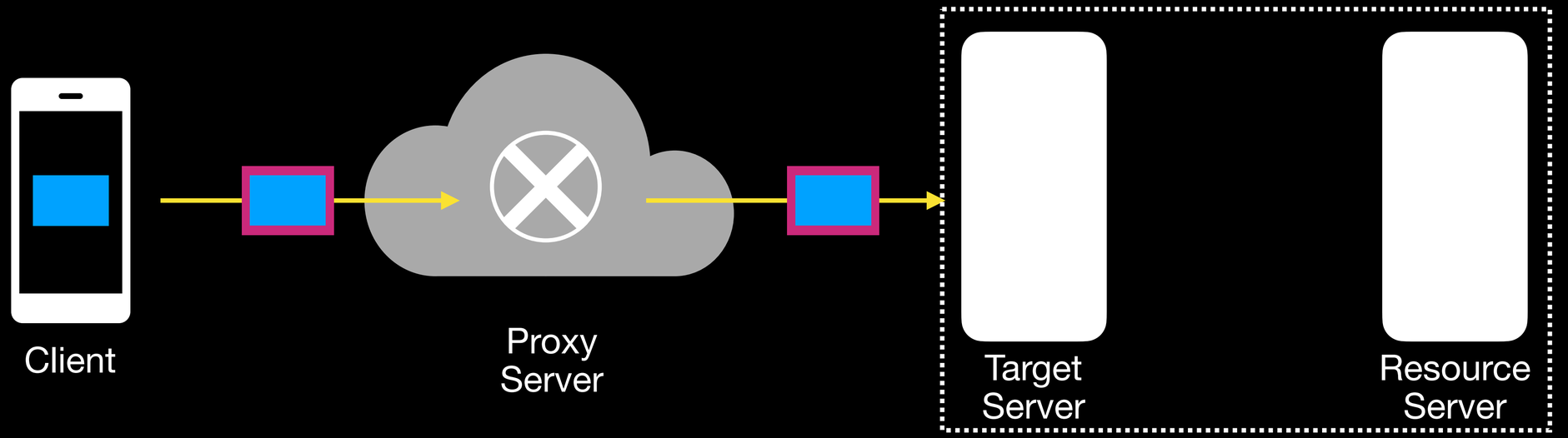
1) Client discovers Target HPKE configuration

Flow
Protocol



2) Client opens HTTPS connection to Proxy

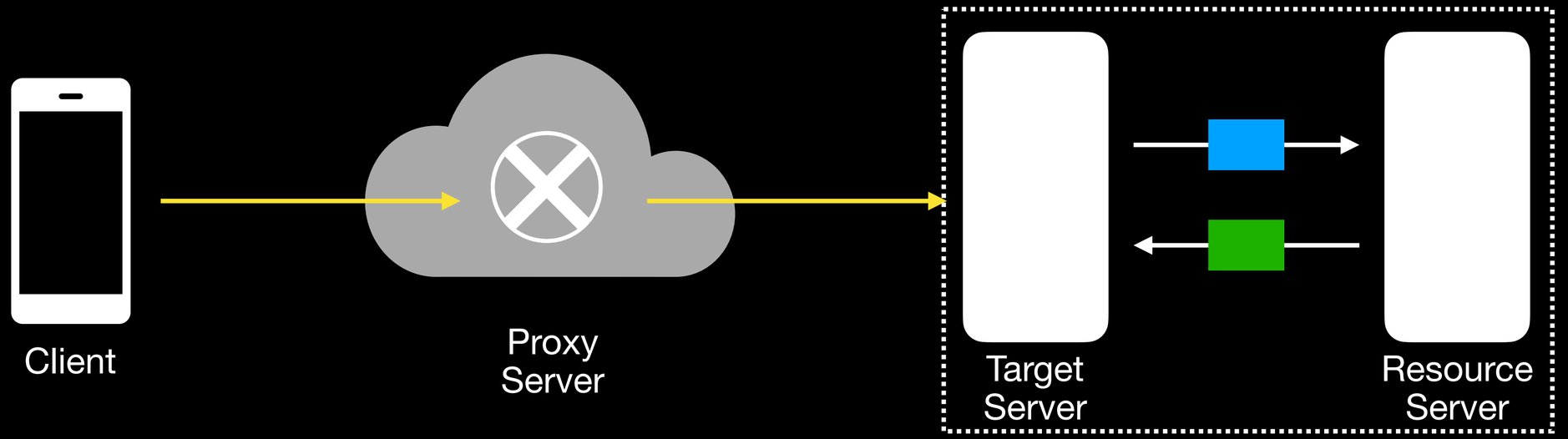
Flow
Protocol



3) Client sends encrypted HTTP message to Target through Proxy

HTTPS
HPKE Encapsulation
HTTP Request

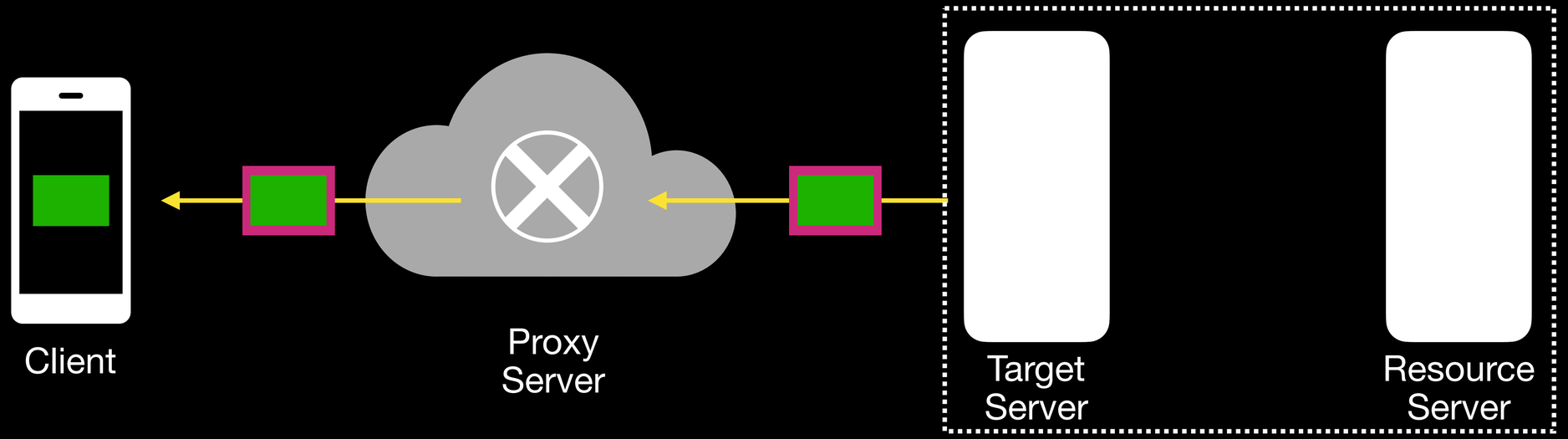
Flow
Protocol



4) Target decapsulates request and sends to resource server and receives response

HTTPS
HPKE Encapsulation
HTTP Request
HTTP Response

Flow
Protocol



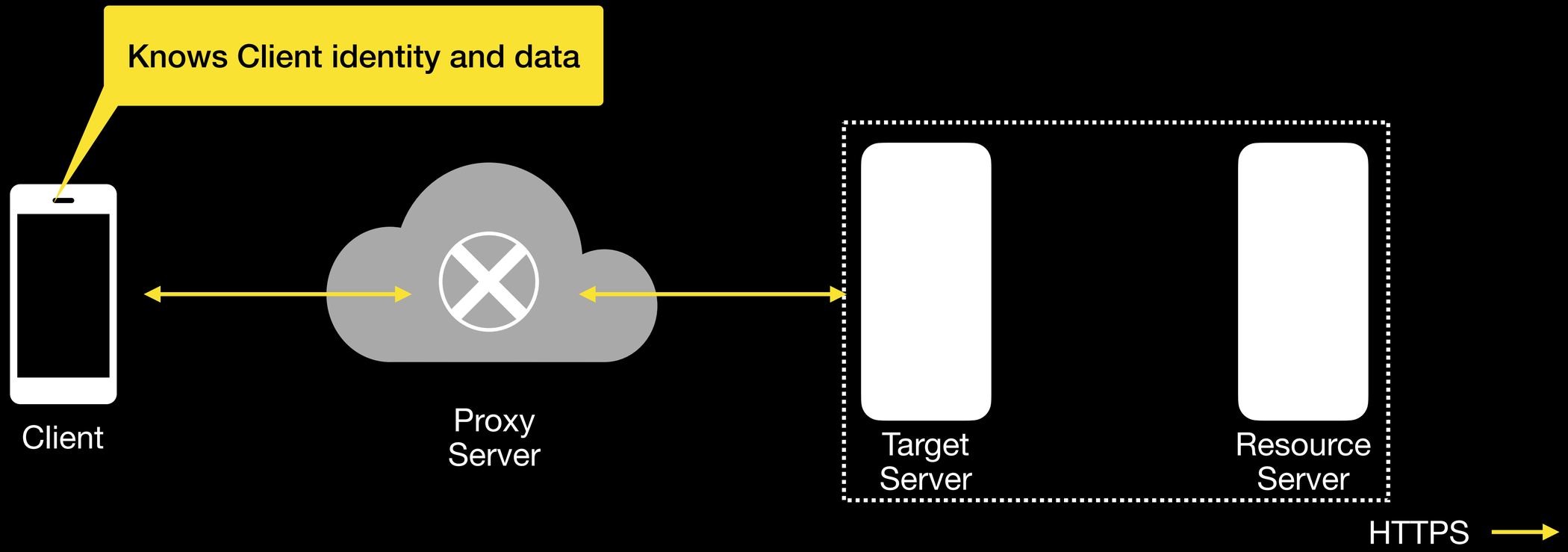
5) Target sends encrypted response to Client through Proxy

HTTPS

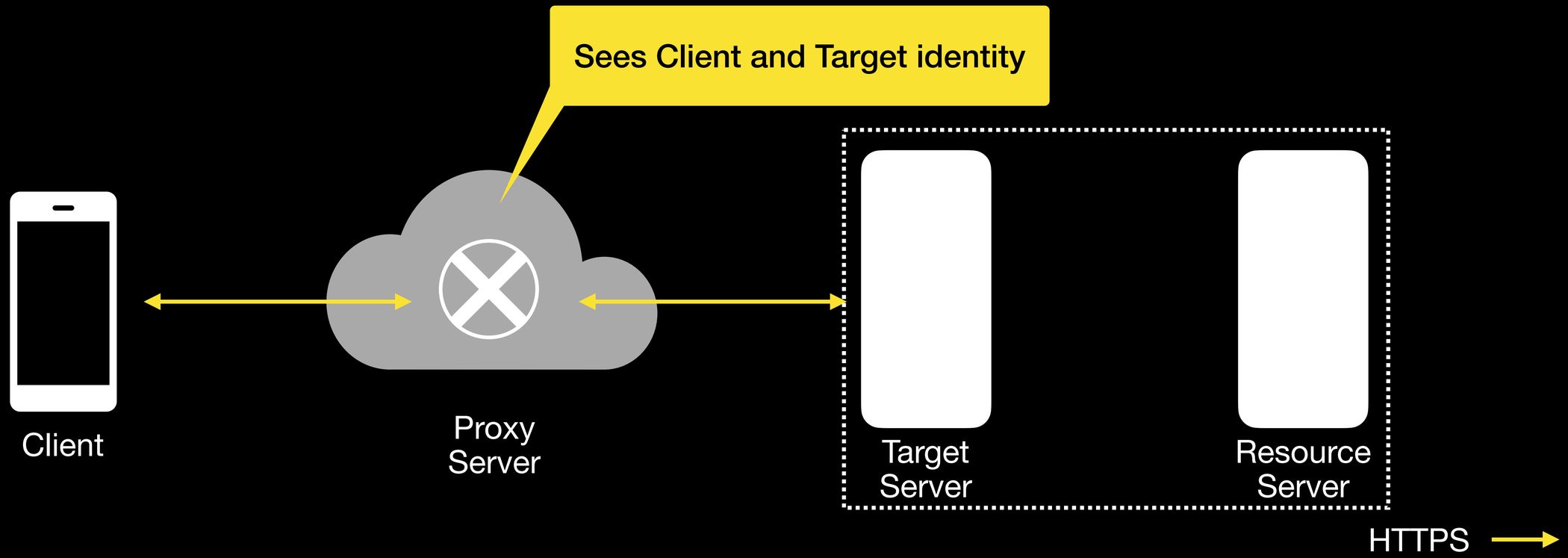
AEAD Encapsulation

HTTP Response

Flow
Protocol



Flow
Protocol



Flow
Protocol

Threat Model

Protocol

The proxy has limited trust from both client and server:

- Client trusts proxy to not leak identity to target
- Target trusts proxy to not overload it

Target compromise allows linking client messages to client identity if target and proxy collude

Traffic analysis (message size features) out of scope

Operational Considerations

Protocol

Limited to specific application protocols such as DNS

- Not general purpose proxy protocol

Proxy and Target configuration *discovery* is out of scope

- Similar to DoH

Proxies need to assist in load management

Targets need to protect against replay attacks from the proxy

Implementations

Running code

Interoperable implementations in Rust and Go

- https://github.com/martinthomson/ohttp
- https://github.com/chris-wood/ohttp-go

Test target available for interop: https://ohttp-echo.crypto-team.workers.dev

Ready for adoption?

Oblivious HTTP

draft-thomson-ohai-ohttp

Thomson, Wood — IETF 112 Online — OHAI

Backup Slides

Open Issues

Protocol details

Streaming request/responses (https://github.com/unicorn-wg/oblivious-http/issues/75)

Additional data (https://github.com/unicorn-wg/oblivious-http/issues/70)

Anti-replay (https://github.com/unicorn-wg/oblivious-http/issues/76)

Shadow banning (https://github.com/unicorn-wg/oblivious-http/issues/66)

Intermediary visibility (https://github.com/unicorn-wg/oblivious-http/issues/61)