Oblivious HTTP: Perfect Forward Security & Key Config Format

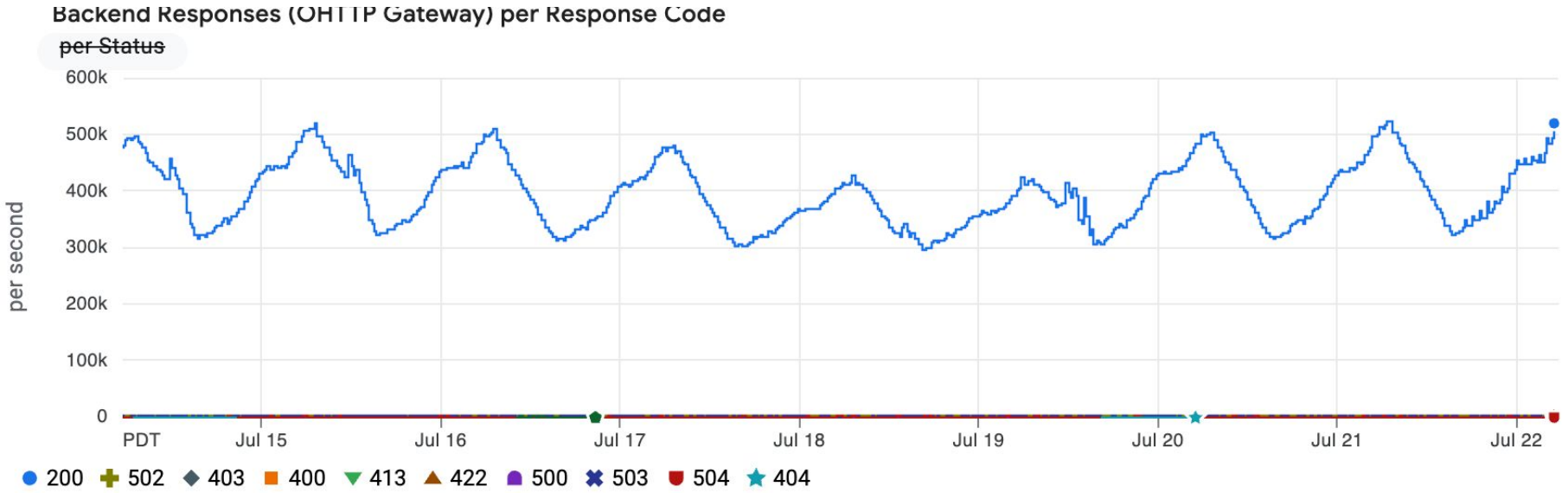
draft-schinazi-httpbis-ohttp-pfs

draft-schinazi-httpbis-ohttp-ext-key-config

HTTPBIS – IETF 126 – Vienna – 2026-07

David Schinazi – dschinazi.ietf@gmail.com

OHTTP go Vroooooooooom




OHAI
I'M BACK

A Perfect Forward Secure Extension to Oblivious HTTP

draft-schinazi-httpbis-ohttp-pfs

In the beginning, there was OHTTP

Then one day, there were chunks

But now the contents are valuable to hackers

Luckily we can solve this with moar cryptography

Crypto! Regular OHTTP

Gateway has stable (pkR, skR)

Client encrypts request via HPKE using pkR

Gateway adds some random entropy for the response

Crypto! PFS OHTTP

Gateway has stable (skR, pkR)

Client encrypts request via HPKE using pkR

Client generates ephemeral (skC, pkC) and sends that to gateway with request

Gateway encrypts response via HPKE using pkC – binding context

If chunked, client switches over to new context

There are plenty of open questions

Do we bind the contexts using the info string?
A PSK?

Are we sure we don't need some entropy?

Do we want this to be backwards compatible with regular OHTTP?

But mainly: is there interest in building PFS OHTTP?


AND NOW FOR
SOMETHING
COMPLETELY
DIFFERENT

An Extensible Key Configuration Format for OHTTP

draft-schinazi-httpbis-ohttp-ext-key-config

But wait, we already have a config format!

But it's got issues

martinthomson commented on Apr 24, 2023 — Collaborator

This changes the definition of "application/ohttp-keys" in a way that is not backwards compatible, but it ensures that the format is usable once new KEMs are introduced.

Length prefix key encoding when there are multiple #284

Merged **chris-wood** merged 5 commits into `main` from `key-config-len` on Jul 26, 2023

Conversation 14 | Commits 5 | Checks 0 | Files changed 1

Changes from **all commits** | **File filter** | **Conversations** | **Jump to**

12 draft-ietf-ohai-ohttp.md

```
@@ -482,12 +482,20 @@ HPKE Symmetric Algorithms:

The "application/ohttp-keys" format is a media type that identifies a serialized
collection of key configurations. The content of this media type comprises one
- or more key configuration encodings (see {{key-config}}) that are concatenated;
- see {{iana-keys}} for a definition of the media type.
+ or more key configuration encodings (see {{key-config}}).  Each encoded
+ configuration is prefixed with a two byte integer in network byte order that
+ indicates the length of the key configuration in bytes.  The length-prefixed
+ encodings are concatenated to form a list.  See {{iana-keys}} for a definition
+ of the media type.
```

OHTTP Keys Rotate!

But as a client you don't know when

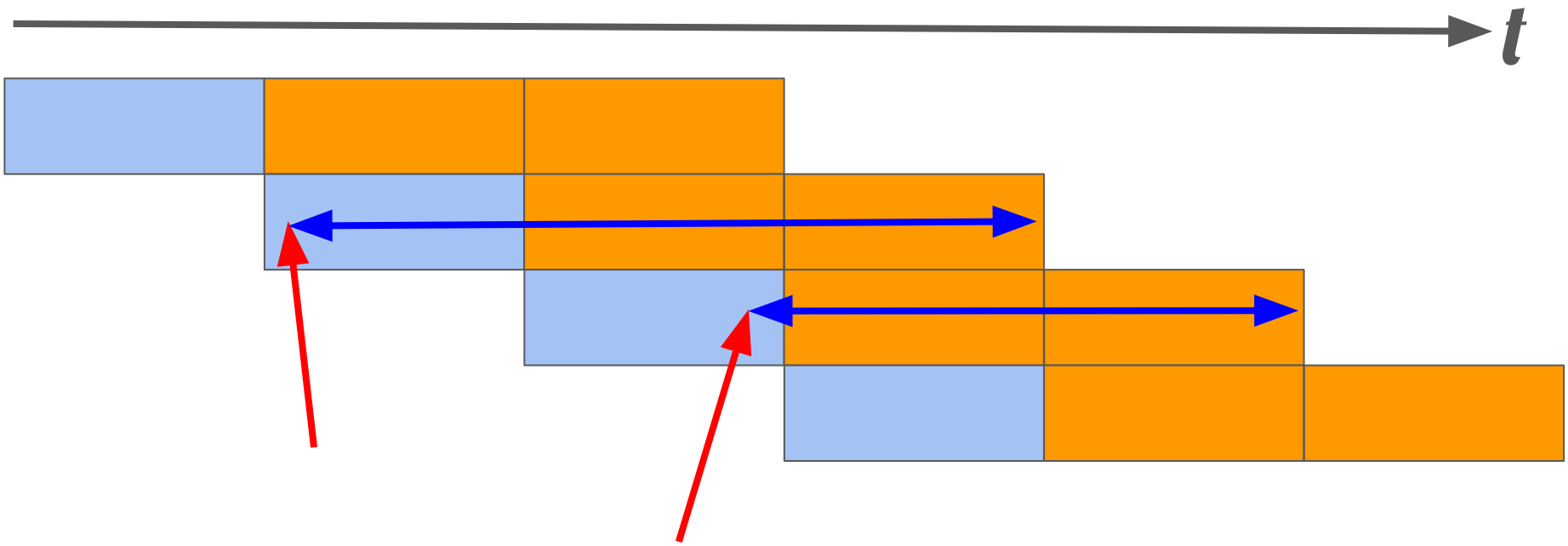
Guidance in RFC 9458 is "IDK, mate. YOLO"

Some clients expire keys after 3 days

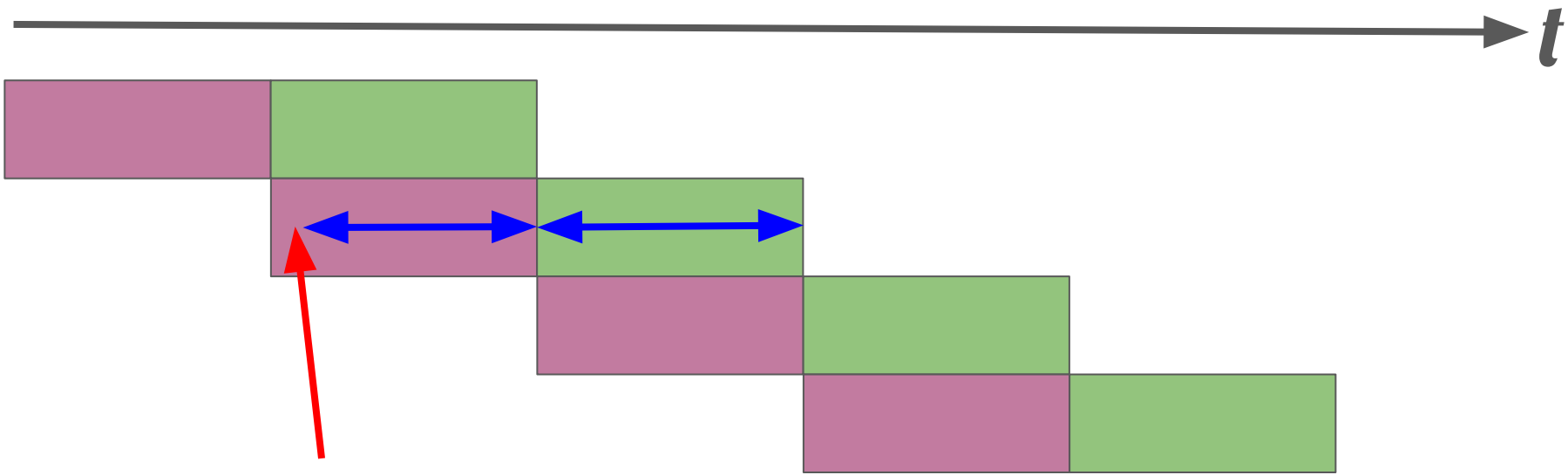
Some clients use keys forever until they get a 4xx

Causes failures in both directions

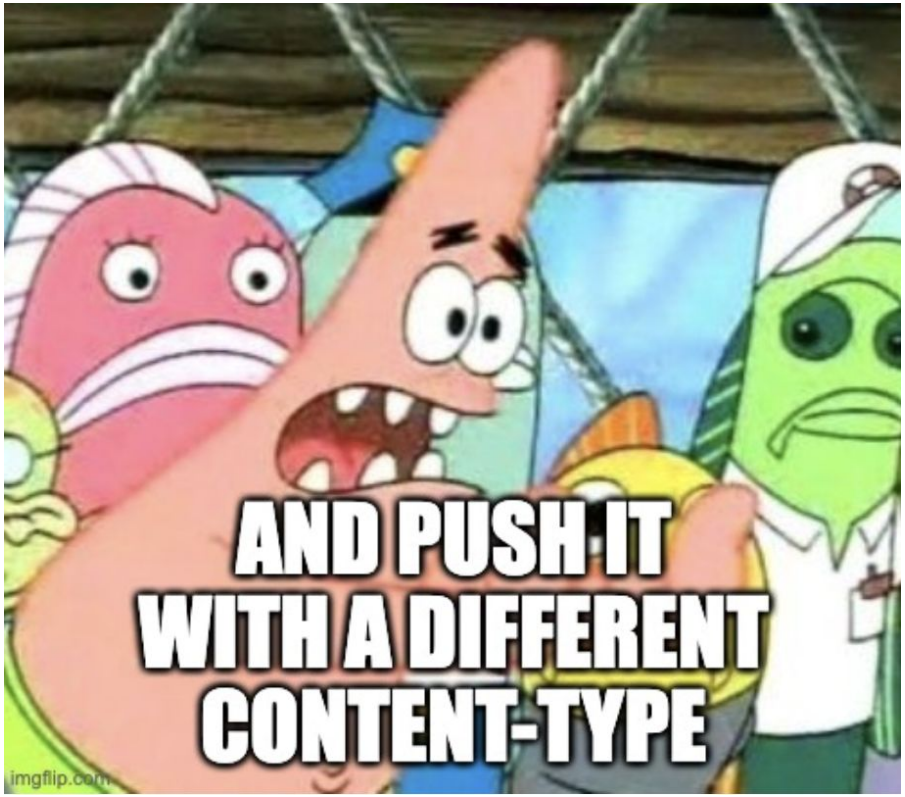
- Keys thrown out prematurely and refetch fails: no key
- Keys used long past expiration: potentially compromised


t

Add Extensions: NOT_BEFORE & EXPIRATION




WE SHOULD
TAKE THE CONFIG


AND PUSH IT
WITH A DIFFERENT
CONTENT-TYPE
imgflip.com

```
Key Config {
    Key Config Length (16),
        Key Identifier (8),
        HPKE KEM ID (16),

            HPKE Public Key (Npk * 8),

        HPKE Symmetric Algorithms Length (16) = 4..65532,

            HPKE Symmetric Algorithms (32) ...,

}
```

```
Key Config {
    Key Config Length (16),
        Key Identifier (8),
        HPKE KEM ID (16),
            HPKE Public Key (Npk * 8),
        HPKE Symmetric Algorithms Length (16) = 4..65532,
            HPKE Symmetric Algorithms (32) ...,
        Extensions (..) ...
}
```