Not so fast!

Otto Moerbeek

PowerDNS Senior Developer

TCP Fast Open?

Stay Open. OX

Agenda

Using TCP Fast Open

01 Why TFO?

02 How does TFO work?

03 Implementing TFO as a server or client

04 TCO in practise: not so fast

05 Conclusions

TCP is becoming more important for DNS

- Larger responses: TXT, DNSSEC, EDNS options
- Enforcing of UDP buffer sizes

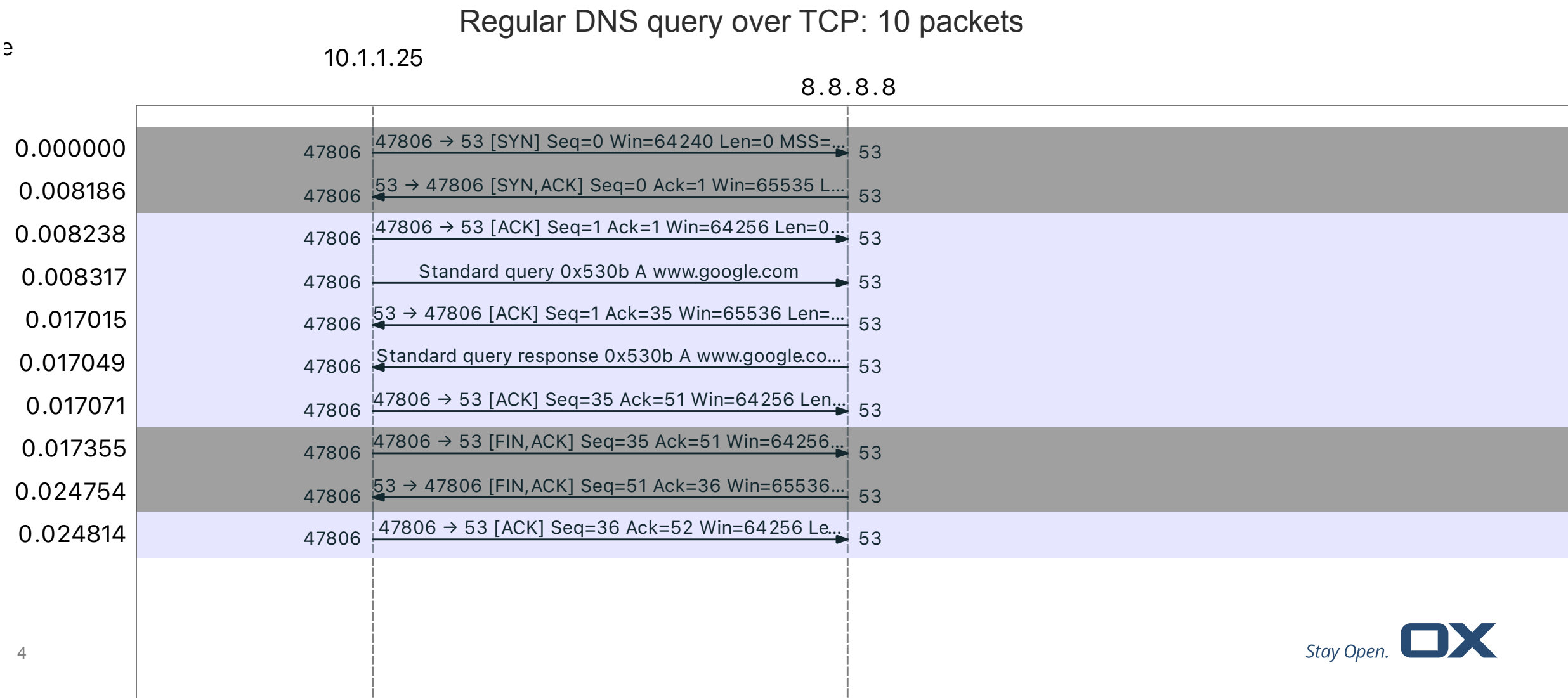
Overhead of TCP is an issue

- Resource consumption on both client and server
- More packets exchanged to get an answer: increased latency

We will look into the last point: overcoming some of the extra latency by using TCP Fast Open (TFO)
At the expense of some more resources used by both client and server

What is TFO?

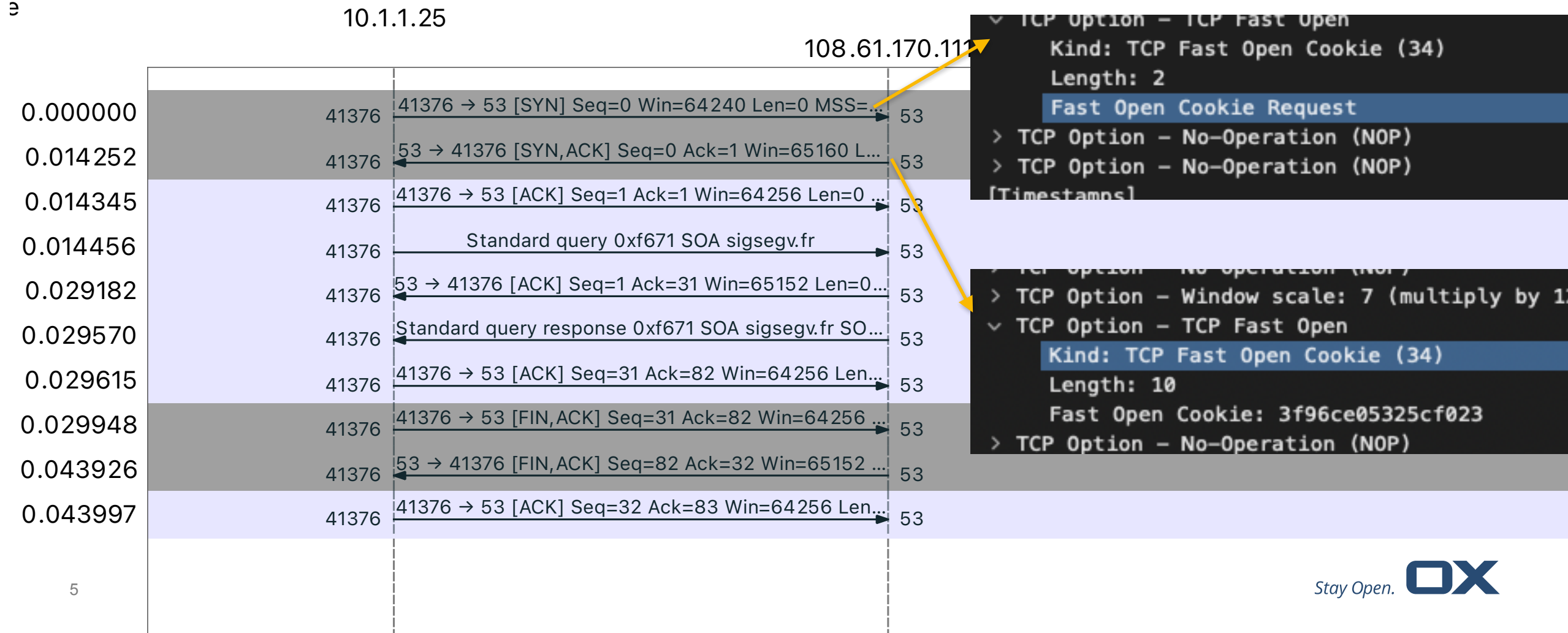
Main purpose: reducing some TCP overhead by exchanging less packets



TFO 1

Client has no cookie, but gets one for future use

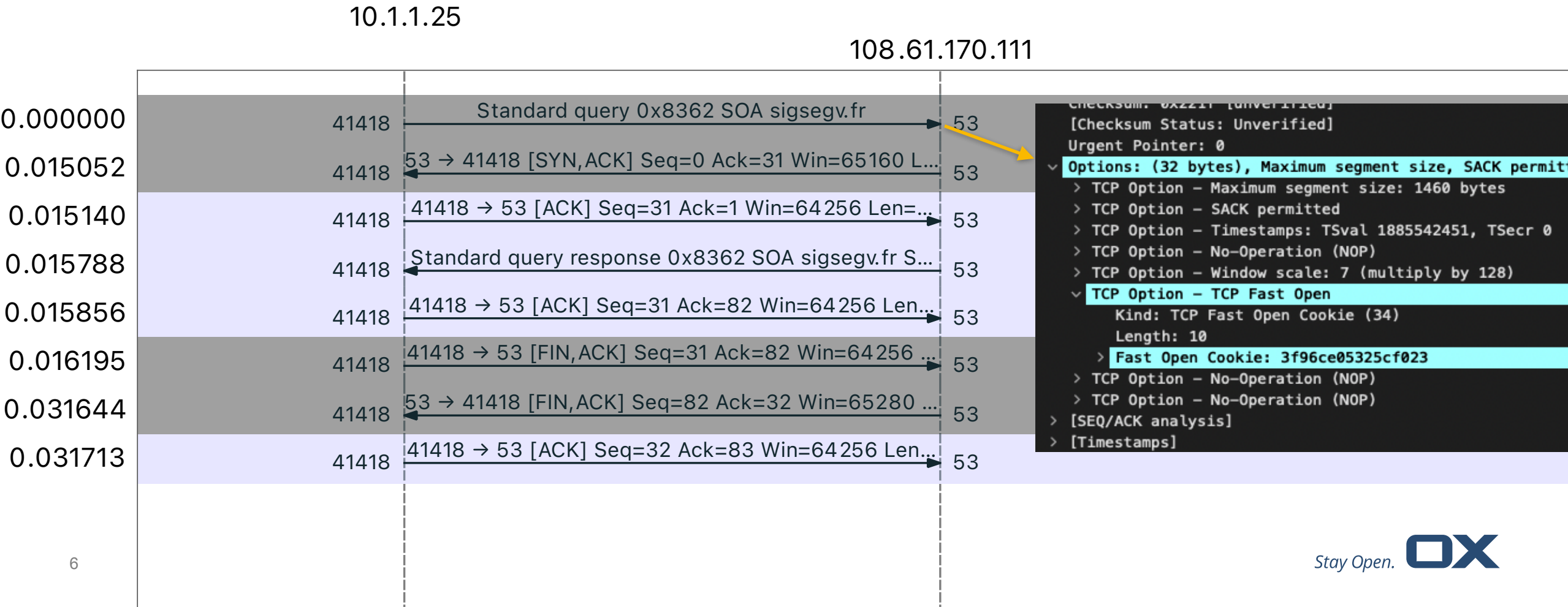
Client indicates cookie support, server sends a cookie



TFO 2

Client has cookie, so we need only 8 packets

Client sends cookie, server accepts cookie
Note the initial packet already has the query



Why the cryptographically secured cookie?

Not using it would open up a heavy form of SYN-flood attack

If the client can do it without proving it did a regular 3WHS before,
it would open up the possibility to flood data into the application
with SYN requests carrying data

Without TFO, the initial SYN can carry data,
but it only gets fed into the app after the 3WHS completes.

A TFO SYN can carry data that immediately gets fed into the app,
without having to wait for the 3WHS to complete

RFC 7413 goes into more details.

Sketch of implementation in OS

Details in RFC 7413

Server side
When a cookie is requested, generated one based on a secret and the client specifics
When a new connection with a cookie is coming in, accept it if it matches

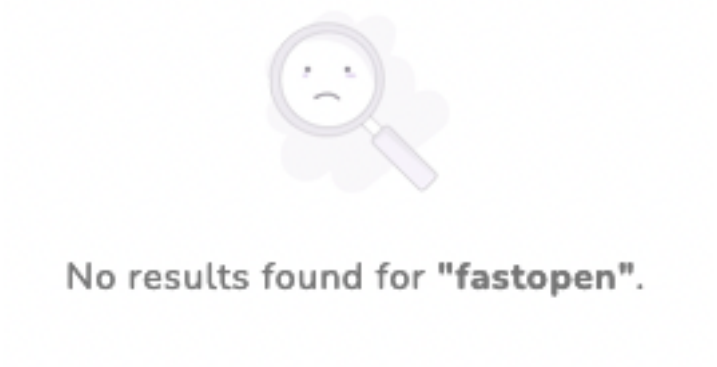
Client side: request cookie if not in store
Save the received cookie, associating it with the the server
Use it (if available) when a new connection is made to the server

Also:
Handle graceful fallback to non-cookie mode
Cleanup old (server/client) cookies

How about TFO support in the DNS world?

It seems pretty well supported by servers

- Most recursive and authoritative servers mention it in release notes
- Not always clear if it is on by default or an option
- For recursive servers it is not always clear if we're talking listening side or connection to auth side
- Stubs? Other clients? Also not very clear…



PowerDNS Recursor already had the server side part, we're going to look at the client side mostly

Server side

As easy as doing a single setsockopt call on the listening socket

But beware, the OS needs some settings as well to enable it

Both server side and client side TFO consume resources

```
sysctl net.ipv4.tcp.fastopen
```

Documented in https://www.kernel.org/doc/html/latest/networking/ip-sysctl.html

Linux has client side enabled by default, *but server side disabled*

This means that on a default install TFO will not be offered to clients, even if it is enabled by the DNS server for on the listening socket

Client side

PowerDNS Recursor was lacking TFO for talking to authoritative servers (i.e. acting as a TCP client)

Enabling it client side can be done with a send(2) flag
Alternatively before calling connect(2) by setting a socket option

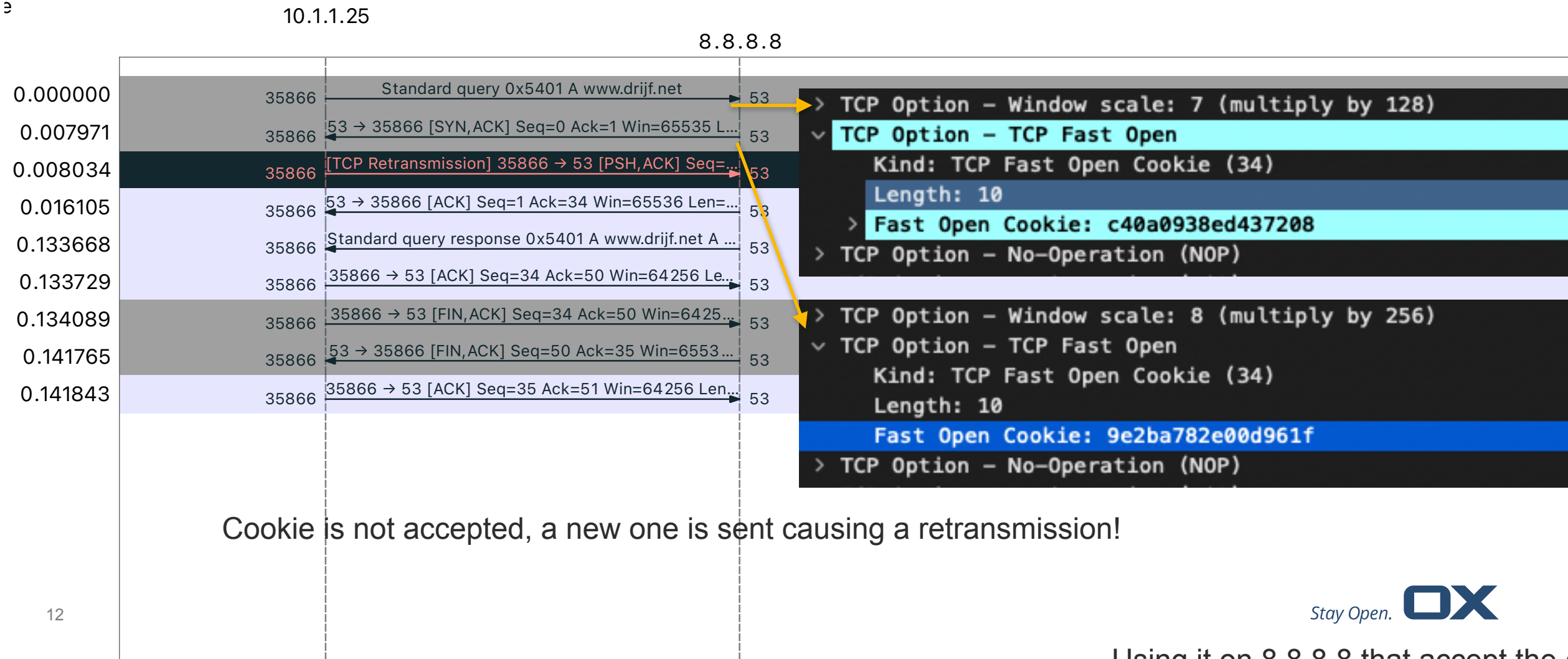
```
setsockopt(s, TCP_FASTOPEN_CONNECT, &on, sizeof(on));
```

Connect calls will return EINPROGRESS if no cookie is available yet
Little code changes (if your code already works with non-blocking sockets)
Does not sound difficult at all.
No real docs found, but the explanation with the PR into Linux explains how to use it…

Hitting problems

Trying it using a command like tool on Quad8

It accepts the client cookie request and returns a cookie as show before, so try to use it for next request:



Cookie is not accepted, a new one is sent causing a retransmission!

Using it on 8.8.8.8 that accept the

Google’s public DNS and TFO

- Quad8 is not the right DNS server to check TFO
- Sends cookies but does not accept them in most cases
- Assumption: cookie is only accepted if you hit the same (anycast) server, which can be forced by fixing the source port, but
- Fixing the source port is not the right thing to do

There are solutions for (anycast) clusters

- Setting a system-wide TFO secret to be shared by cluster members (`sysctl net.ipv4.tcp_fastopen_key`)
- Use a socket option to set a socket specific TFO secret (`setsockopt(… TCP_FASTOPEN_KEY …)`)

- Both not implemented at Quad8!
- This is a known issue since 2017 at Google: https://issuetracker.google.com/68040969
- Sad fact: both the TFO RFC and the Linux implementation originate from Google
- Also observed at Google's authoritative servers
- But: Google's DoT and DoH offer and accept TFO correctly

Cloudflare, Quad9 and Authoritative servers?

- At time of writing, both 1.1.1.1 and 9.9.9.9 support TFO (only) for DoH

How about authoritative servers as found in the wild?

- Support by authoritative servers is rare
- In *DNS Privacy in Practice and Preparation,* Deccio & Davis mention between 1% (Alexa5000) and 3% (TLDs)
- This is data from 2019, it would be nice to repeat the study

Testing is hard

Often TFO cookies would stop working

Captures showed that cookies received previously were not used for new connections
Turns out that Linux is extremely conservative in the usage of TFO as a client

Some counters as shown by `netstat -s`:

```
$ netstat -s | grep TCPF
    TCPFastOpenActive: 1
    TCPFastOpenActiveFail: 3
    TCPFastOpenBlackhole: 1
```

There are a few more. I could not find documentation for these counters.
But reading code showed that a single timeout on connect is already enough to stop using TFO for *all* client connections for `net.ipv4.tcp_fastopen_blackhole_timeout_sec`, with exponential backoff to even longer periods. Default is 3600s…

Remedy: set the sysctl to a lower value, maybe even 0.

Conclusions

TFO may be fast, but adoption is slow

- TFO is nice and not hard to implement from the application developer point of view, the OS does all the work
- Linux fails to document the socket options and counters properly, making it harder to deploy than necessary
- Docs for various DNS servers and client libs are also not clear most of the time
- Sadly, even a large party that should have all the resources and expertise fails to do it properly server side
- On the client side, a single mishap has big consequences: disabling TFO for a long time for all connections

These things need to change to increase TFO adoption.

References

- Deccio and Davis: “DNS Privacy in Practice and Preparation”
- RFC 7413