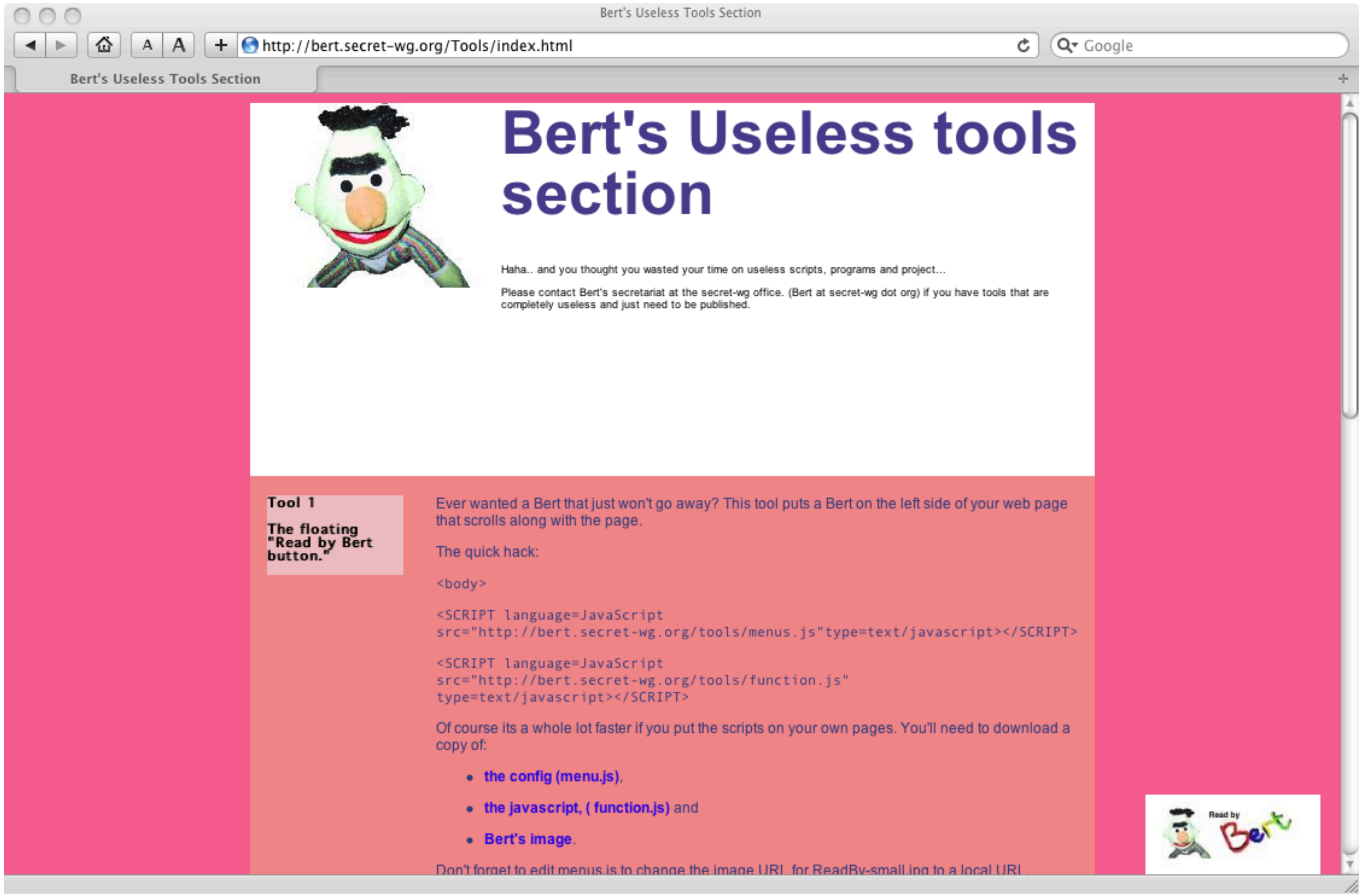
Stateless ~~Dateless~~ and DNS ~~Desperate~~ !

Geoff Huston

APNIC


Bert's Useless Tools Section
http://bert.secret-wg.org/Tools/index.html
Google
Bert's Useless Tools Section
Read by Bert

Bert's Useless tools section

Haha.. and you thought you wasted your time on useless scripts, programs and project...

Please contact Bert's secretariat at the secret-wg office. (Bert at secret-wg dot org) if you have tools that are completely useless and just need to be published.

Tool 1

The floating "Read by Bert button."

Ever wanted a Bert that just won't go away? This tool puts a Bert on the left side of your web page that scrolls along with the page.

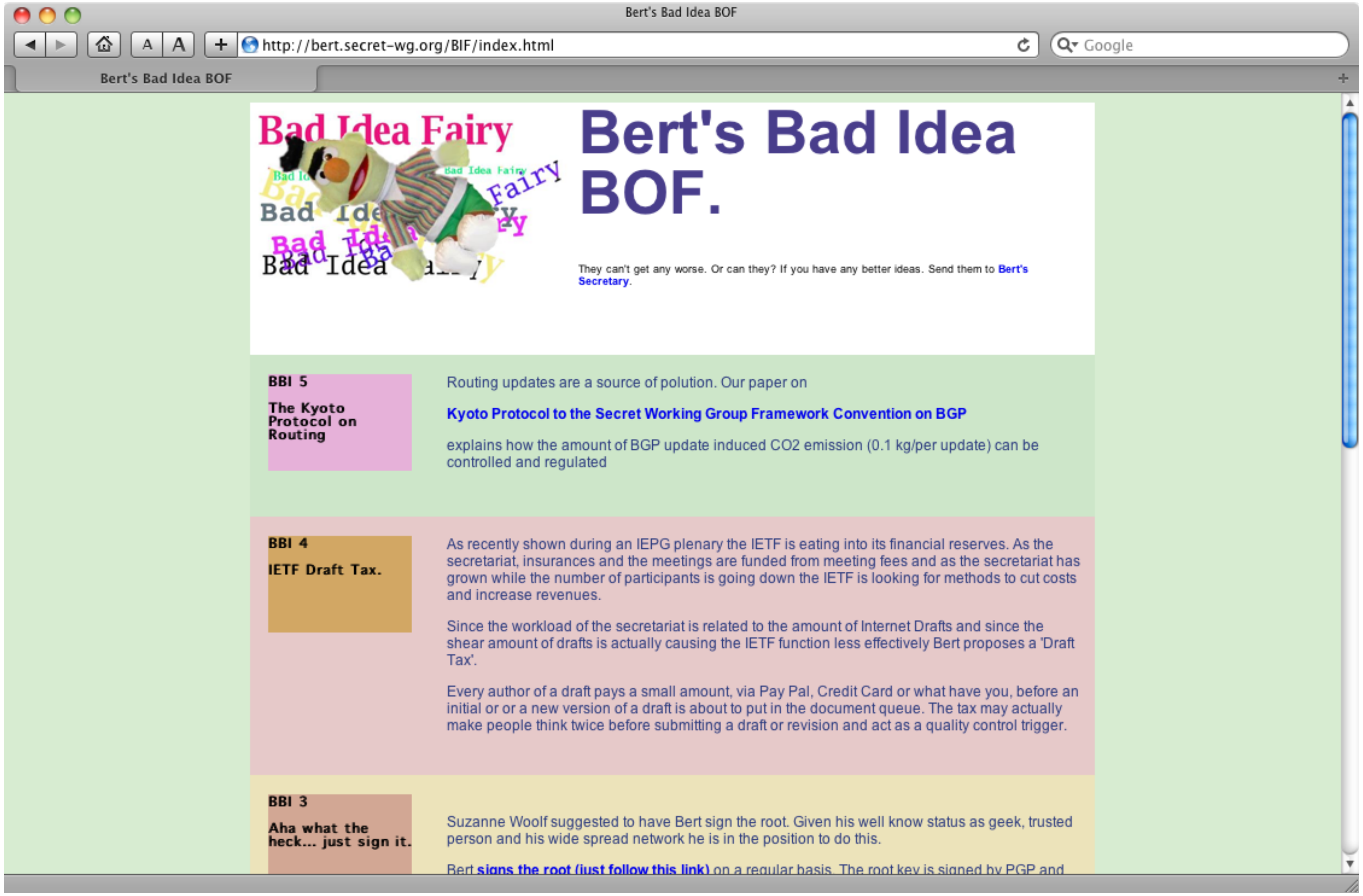
The quick hack:

```
<body>

<SCRIPT language=JavaScript
src="http://bert.secret-wg.org/tools/menus.js"type=text/javascript></SCRIPT>

<SCRIPT language=JavaScript
src="http://bert.secret-wg.org/tools/function.js"
type=text/javascript></SCRIPT>
```

Of course its a whole lot faster if you put the scripts on your own pages. You'll need to download a copy of:

- the config (menu.js),
- the javascript, (function.js) and
- Bert's image.

Don't forget to edit menus.js to change the image URL for ReadBy-small.jpg to a local URL.



Bert's Bad Idea BOF

http://bert.secret-wg.org/BIF/index.html

Bert's Bad Idea BOF.

They can't get any worse. Or can they? If you have any better ideas. Send them to **Bert's Secretary**.

BBI 5

The Kyoto Protocol on Routing

Routing updates are a source of polution. Our paper on

Kyoto Protocol to the Secret Working Group Framework Convention on BGP

explains how the amount of BGP update induced CO2 emission (0.1 kg/per update) can be controlled and regulated

BBI 4

IETF Draft Tax.

As recently shown during an IEPG plenary the IETF is eating into its financial reserves. As the secretariat, insurances and the meetings are funded from meeting fees and as the secretariat has grown while the number of participants is going down the IETF is looking for methods to cut costs and increase revenues.

Since the workload of the secretariat is related to the amount of Internet Drafts and since the shear amount of drafts is actually causing the IETF function less effectively Bert proposes a 'Draft Tax'.

Every author of a draft pays a small amount, via Pay Pal, Credit Card or what have you, before an initial or or a new version of a draft is about to put in the document queue. The tax may actually make people think twice before submitting a draft or revision and act as a quality control trigger.

BBI 3

Aha what the heck... just sign it.

Suzanne Woolf suggested to have Bert sign the root. Given his well know status as geek, trusted person and his wide spread network he is in the position to do this.

Bert **signs the root (just follow this link)** on a regular basis. The root key is signed by PGP and

Can I do both at once?

This is definitely a Bad Idea ...

with that intriguing possibility that it just might be made to work

making it a Useless Tool at the same time!

IP Networking 101

- There are two major transport protocols:
  - TCP when reliable data transfer is needed
  - UDP for simple lightweight transactions

IP Networking 102

Client / Server Transaction Support

- TCP has limitations
  - server load, connection intensity limitations, vulnerability to TCP SYN and RST attacks
- UDP has limitations
  - requires IP fragmentation handling for large UDP packets
  - and just how does IPv6 handle UDP fragmentation when the effective path MTU is less than the interface MTU?
    - i.e, Q: how does DNS on UDP on IPv6 work when there are path MTU constraints lower than the local MTU? A: It doesn't!

IP Networking 102

Coping with large responses – what happens when the response size exceeds the path MTU?

- Use UDP with IP level fragmentation and reassembly?
  - but firewalls often drop trailing IP fragments
  - IPv6 UDP path MTU handling is not well suited to transaction apps
- Use TCP segmentation andreassembly?
  - switching to TCP implies additional load on the server, limitations on server query capacity, and additional delay in the elapsed time for the transaction

IP Networking 666

Lets fire up the Bad Idea Factory!

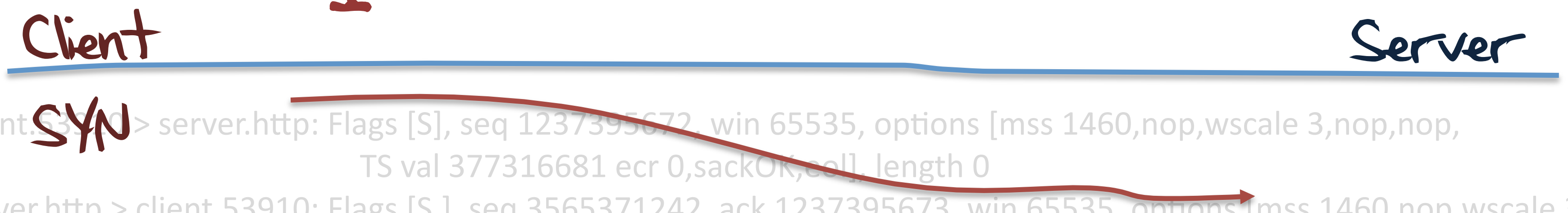
Why not combine UDP with TCP segmentation and reassembly?

- The client runs a conventional TCP application
- The server runs a stateless UDP-style application, but formats its output using TCP framing
- i.e. What the server wants is "Stateless TCP"

A Simple TCP Transaction

```
client.53910 > server.http: Flags [S], seq 1237395672, win 65535, options [mss 1460,nop,wscale 3,nop,nop,
                            TS val 377316681 ecr 0,sackOK,eol], length 0
server.http > client.53910: Flags [S.], seq 3565371242, ack 1237395673, win 65535, options [mss 1460,nop,wscale 3,
                            sackOK,TS val 2295904142 ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142], length 0
client.53910 > server.http: Flags [P.], seq 1:248, ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142],
                            length 247
server.http > client.53910: Flags [P.], seq 1:468, ack 248, win 8326, options [nop,nop,TS val 2295904189
                            ecr 377316681], length 467
server.http > client.53910: Flags [F.], seq 468, ack 248, win 8326, options [nop,nop,TS val 2295904189
                            ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 468, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
                             length 0
client.53910 > server.http: Flags [.], ack 469, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
                            length 0
client.53910 > server.http: Flags [F.], seq 248, ack 469, win 65535, options [nop,nop,TS val 377316682
                            ecr 2295904189], length 0
server.http > client.53910: Flags [.], ack 249, win 8325, options [nop,nop,TS val 2295904237 ecr 377316682], length 0
```

A Simple TCP Transaction



```
client.53910 > server.http: Flags [S], seq 1237395672, win 65535, options [mss 1460,nop,wscale 3,nop,nop,
                        TS val 377316681 ecr 0,sackOK,eol], length 0
server.http > client.53910: Flags [S.], seq 3565371242, ack 1237395673, win 65535, options [mss 1460,nop,wscale 3,
                        sackOK,TS val 2295904142 ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142], length 0
client.53910 > server.http: Flags [P.], seq 1:248, ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142],
                        length 247
server.http > client.53910: Flags [P.], seq 1:468, ack 248, win 8326, options [nop,nop,TS val 2295904189
                        ecr 377316681], length 467
server.http > client.53910: Flags [F.], seq 468, ack 248, win 8326, options [nop,nop,TS val 2295904189
                        ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 468, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
                         length 0
client.53910 > server.http: Flags [.], ack 469, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
                        length 0
client.53910 > server.http: Flags [F.], seq 248, ack 469, win 65535, options [nop,nop,TS val 377316682
                        ecr 2295904189], length 0
server.http > client.53910: Flags [.], ack 249, win 8325, options [nop,nop,TS val 2295904237 ecr 377316682], length 0
```

A Simple TCP Transaction

Client

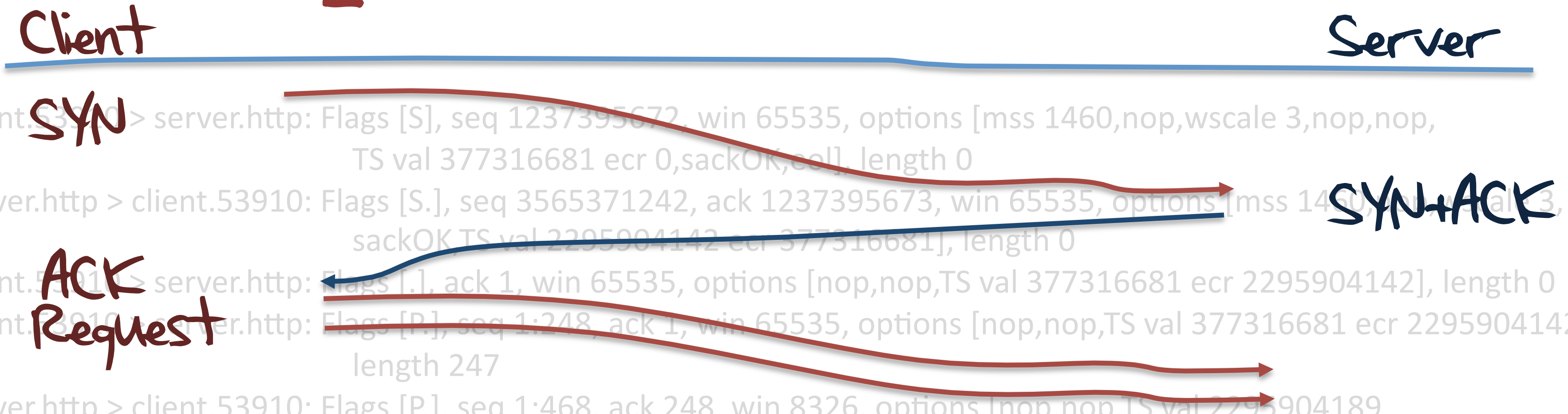
Server

SYN

SYN+ACK

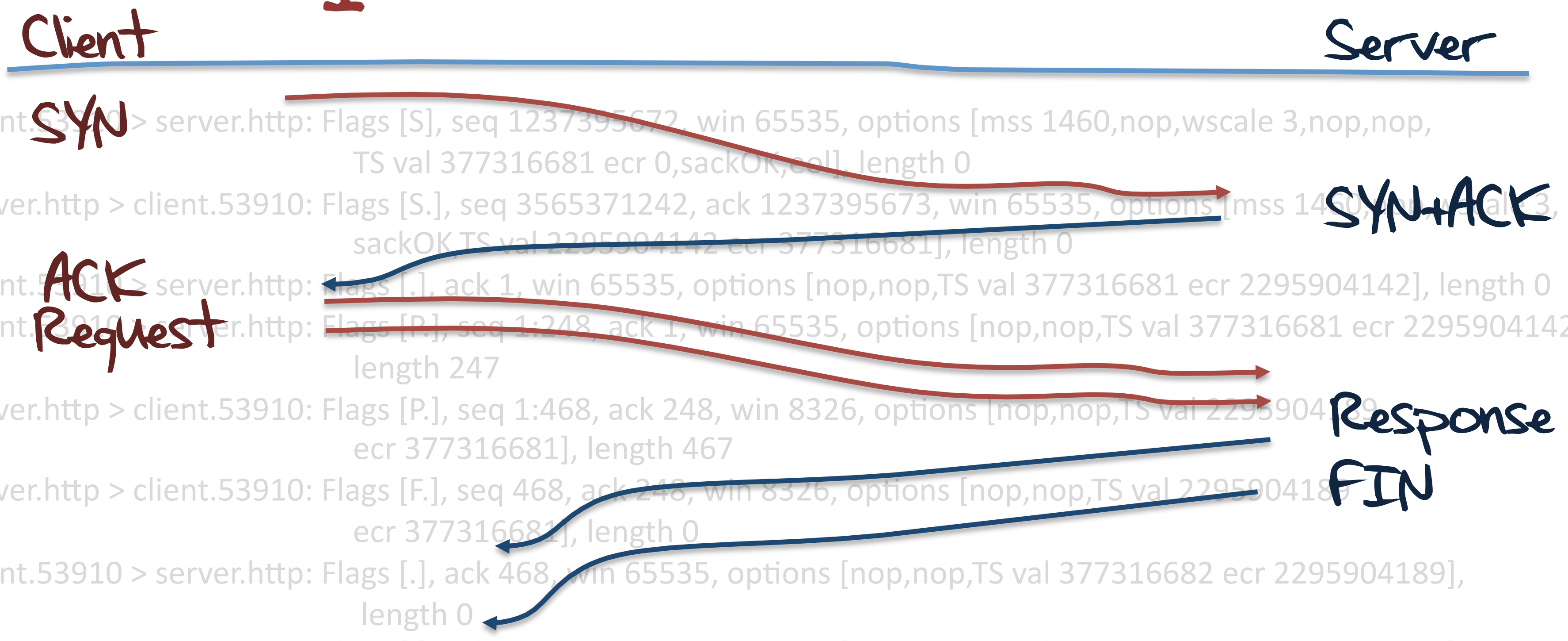
```
client.53910 > server.http: Flags [S], seq 1237395672, win 65535, options [mss 1460,nop,wscale 3,nop,nop,
                        TS val 377316681 ecr 0,sackOK,eol], length 0
server.http > client.53910: Flags [S.], seq 3565371242, ack 1237395673, win 65535, options [mss 1460,nop,wscale 3,
                        sackOK,TS val 2295904142 ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142], length 0
client.53910 > server.http: Flags [P.], seq 1:248, ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142],
                        length 247
server.http > client.53910: Flags [P.], seq 1:468, ack 248, win 8326, options [nop,nop,TS val 2295904189
                        ecr 377316681], length 467
server.http > client.53910: Flags [F.], seq 468, ack 248, win 8326, options [nop,nop,TS val 2295904189
                        ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 468, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
                         length 0
client.53910 > server.http: Flags [.], ack 469, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
                        length 0
client.53910 > server.http: Flags [F.], seq 248, ack 469, win 65535, options [nop,nop,TS val 377316682
                        ecr 2295904189], length 0
server.http > client.53910: Flags [.], ack 249, win 8325, options [nop,nop,TS val 2295904237 ecr 377316682], length 0
```

A Simple TCP Transaction



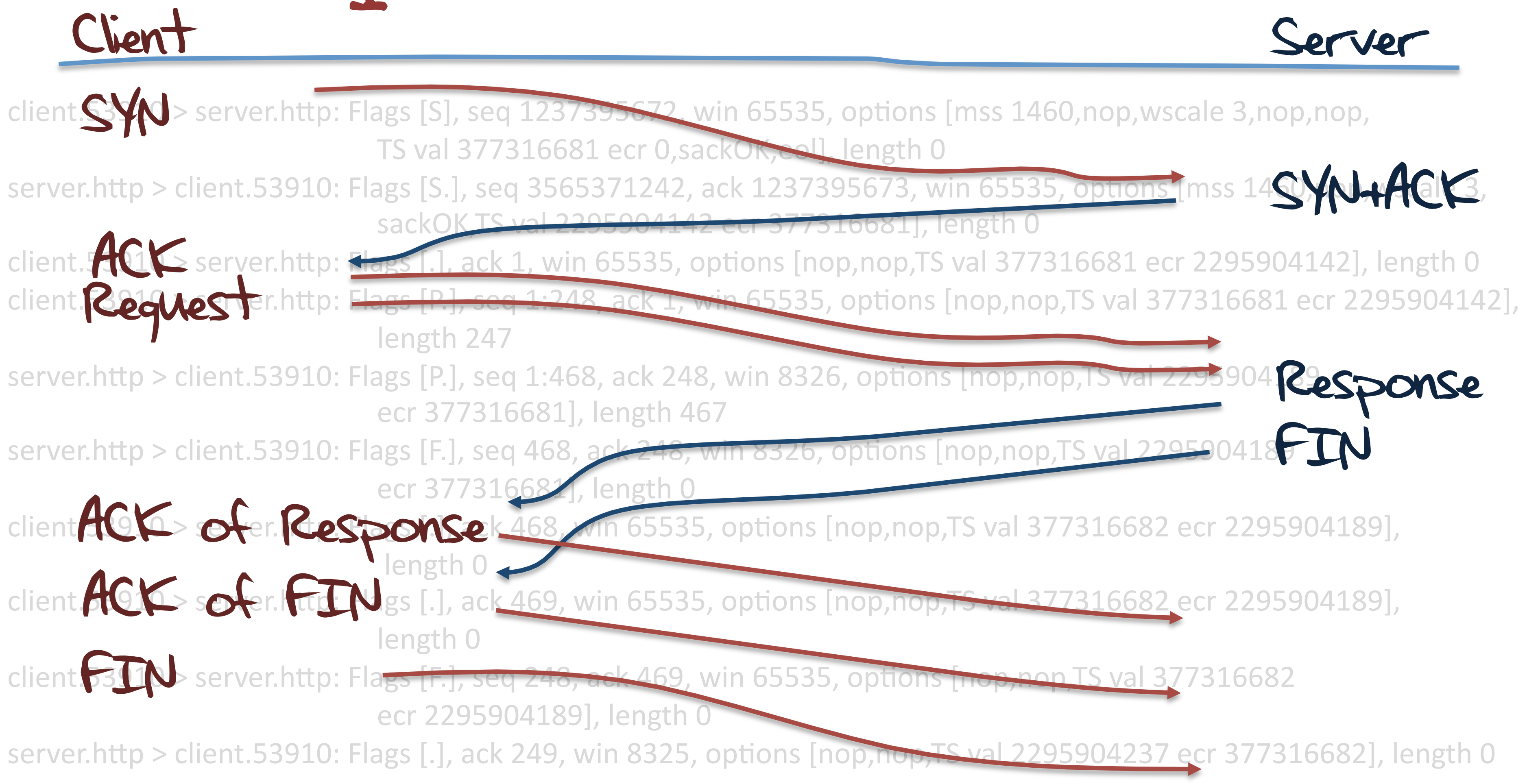
```
client.53910 > server.http: Flags [S], seq 1237395672, win 65535, options [mss 1460,nop,wscale 3,nop,nop,
                                  TS val 377316681 ecr 0,sackOK,eol], length 0
server.http > client.53910: Flags [S.], seq 3565371242, ack 1237395673, win 65535, options [mss 1460,nop,wscale 3,
                                  sackOK,TS val 2295904142 ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142], length 0
client.53910 > server.http: Flags [P.], seq 1:248, ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142],
                                  length 247
server.http > client.53910: Flags [P.], seq 1:468, ack 248, win 8326, options [nop,nop,TS val 2295904189
                                  ecr 377316681], length 467
server.http > client.53910: Flags [F.], seq 468, ack 248, win 8326, options [nop,nop,TS val 2295904189
                                  ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 468, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
                                   length 0
client.53910 > server.http: Flags [.], ack 469, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
                                  length 0
client.53910 > server.http: Flags [F.], seq 248, ack 469, win 65535, options [nop,nop,TS val 377316682
                                  ecr 2295904189], length 0
server.http > client.53910: Flags [.], ack 249, win 8325, options [nop,nop,TS val 2295904237 ecr 377316682], length 0
```

A Simple TCP Transaction



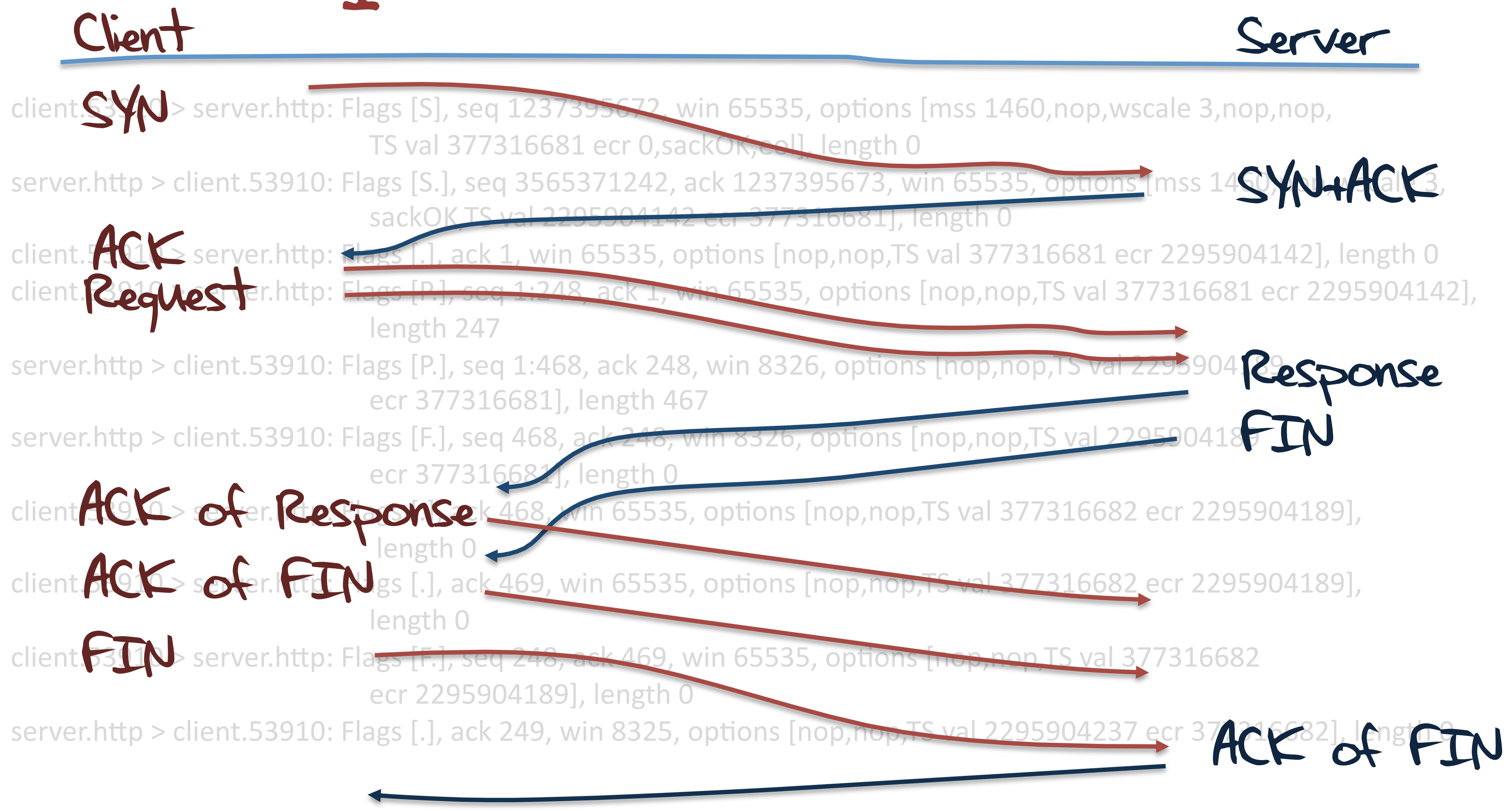
```
client.53910 > server.http: Flags [S], seq 1237395672, win 65535, options [mss 1460,nop,wscale 3,nop,nop,
                 TS val 377316681 ecr 0,sackOK,eol], length 0
server.http > client.53910: Flags [S.], seq 3565371242, ack 1237395673, win 65535, options [mss 1460,nop,wscale 3,
                 sackOK,TS val 2295904142 ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142], length 0
client.53910 > server.http: Flags [P.], seq 1:248, ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142],
                 length 247
server.http > client.53910: Flags [P.], seq 1:468, ack 248, win 8326, options [nop,nop,TS val 2295904189
                 ecr 377316681], length 467
server.http > client.53910: Flags [F.], seq 468, ack 248, win 8326, options [nop,nop,TS val 2295904189
                 ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 468, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
                  length 0
client.53910 > server.http: Flags [.], ack 469, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
                 length 0
client.53910 > server.http: Flags [F.], seq 248, ack 469, win 65535, options [nop,nop,TS val 377316682
                 ecr 2295904189], length 0
server.http > client.53910: Flags [.], ack 249, win 8325, options [nop,nop,TS val 2295904237 ecr 377316682], length 0
```

A Simple TCP Transaction



```
client.53910 > server.http: Flags [S], seq 1237395672, win 65535, options [mss 1460,nop,wscale 3,nop,nop,
            TS val 377316681 ecr 0,sackOK,eol], length 0
server.http > client.53910: Flags [S.], seq 3565371242, ack 1237395673, win 65535, options [mss 1460,nop,wscale 3,
            sackOK,TS val 2295904142 ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142], length 0
client.53910 > server.http: Flags [P.], seq 1:248, ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142],
            length 247
server.http > client.53910: Flags [P.], seq 1:468, ack 248, win 8326, options [nop,nop,TS val 2295904189
            ecr 377316681], length 467
server.http > client.53910: Flags [F.], seq 468, ack 248, win 8326, options [nop,nop,TS val 2295904189
            ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 468, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
            length 0
client.53910 > server.http: Flags [.], ack 469, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
            length 0
client.53910 > server.http: Flags [F.], seq 248, ack 469, win 65535, options [nop,nop,TS val 377316682
            ecr 2295904189], length 0
server.http > client.53910: Flags [.], ack 249, win 8325, options [nop,nop,TS val 2295904237 ecr 377316682], length 0
```

A Simple TCP Transaction



```
client.53910 > server.http: Flags [S], seq 1237395672, win 65535, options [mss 1460,nop,wscale 3,nop,nop,
            TS val 377316681 ecr 0,sackOK,eol], length 0
server.http > client.53910: Flags [S.], seq 3565371242, ack 1237395673, win 65535, options [mss 1460,nop,wscale 3,
            sackOK,TS val 2295904142 ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142], length 0
client.53910 > server.http: Flags [P.], seq 1:248, ack 1, win 65535, options [nop,nop,TS val 377316681 ecr 2295904142],
            length 247
server.http > client.53910: Flags [P.], seq 1:468, ack 248, win 8326, options [nop,nop,TS val 2295904189
            ecr 377316681], length 467
server.http > client.53910: Flags [F.], seq 468, ack 248, win 8326, options [nop,nop,TS val 2295904189
            ecr 377316681], length 0
client.53910 > server.http: Flags [.], ack 468, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
            length 0
client.53910 > server.http: Flags [.], ack 469, win 65535, options [nop,nop,TS val 377316682 ecr 2295904189],
            length 0
client.53910 > server.http: Flags [F.], seq 248, ack 469, win 65535, options [nop,nop,TS val 377316682
            ecr 2295904189], length 0
server.http > client.53910: Flags [.], ack 249, win 8325, options [nop,nop,TS val 2295904237 ecr 377316682], length 0
```

The Server's Perspective

1. SYN Response

$$\begin{tikzcd}
\text{SYN} \arrow[dr] & \\
& \text{Server} \arrow[dl] \\
\text{SYN+ACK} &
\end{tikzcd}$$

Flip the IP source and destination fields

Flip the TCP source and destination ports

Use any old sequence number

Offer a reasonable MSS (1220)

Offer no other TCP options

The Server's Perspective

2. Request Response

$$\begin{tikzcd} \text{Request} \arrow[r] & \text{Server} \arrow[dl] \arrow[ddl] \\ \text{Response} & \\ \text{FIN} & \end{tikzcd}$$

Send an ACK

Start with a sequence number given in the Request ACK

Generate the response

Chop the response into 512 octet segments

Send the segment train back to back

Send a FIN

The Server's Perspective

3. FIN Response

$$\begin{tikzcd} \text{FIN} \arrow[r] & \text{Server} \arrow[dl, bend left] \\ \text{ACK} & \end{tikzcd}$$

Flip the IP addrs, TCP ports and ack/sequence fields

increment ack field

send ACK

The Stateless TCP Server's Perspective

1. SYN

$$\begin{tikzcd}
\text{SYN} \arrow[r] & \text{Stateless Server} \arrow[dl] \\
\text{SYN+ACK} &
\end{tikzcd}$$

2. Request (payload)

$$\begin{tikzcd}
\text{Request} \arrow[r] & \text{Stateless Server} \arrow[dl] \arrow[ddl] \\
\text{Response} & \\
\text{FIN} &
\end{tikzcd}$$

3. FIN

$$\begin{tikzcd}
\text{FIN} \arrow[r] & \text{Stateless Server} \arrow[dl] \\
\text{ACK} &
\end{tikzcd}$$

4. all else

all else – no Server response

Can this be coded?

How can a user application see raw IP packets without the kernel getting in the way?

libpcap is one option:

```
/*associate a device to the packet capture process */
pcap_lookupnet(dev, &net, &mask, errbuff) ;

/* open capture device */
handle = pcap_open_live(dev, SNAP_LEN, 1, 1000, errbuff);

/* set the capture filter e*/
filter_exp = "dst port 80 and dst host 192.0.2.1";
pcap_compile(handle, &fp, filter_exp, 0, net) ;
pcap_setfilter(handle, &fp) ;

/* and start it up , sending raw packets to got_packet() routine */
pcap_loop(handle, -1, got_packet, NULL) ;
```

Can this be coded?

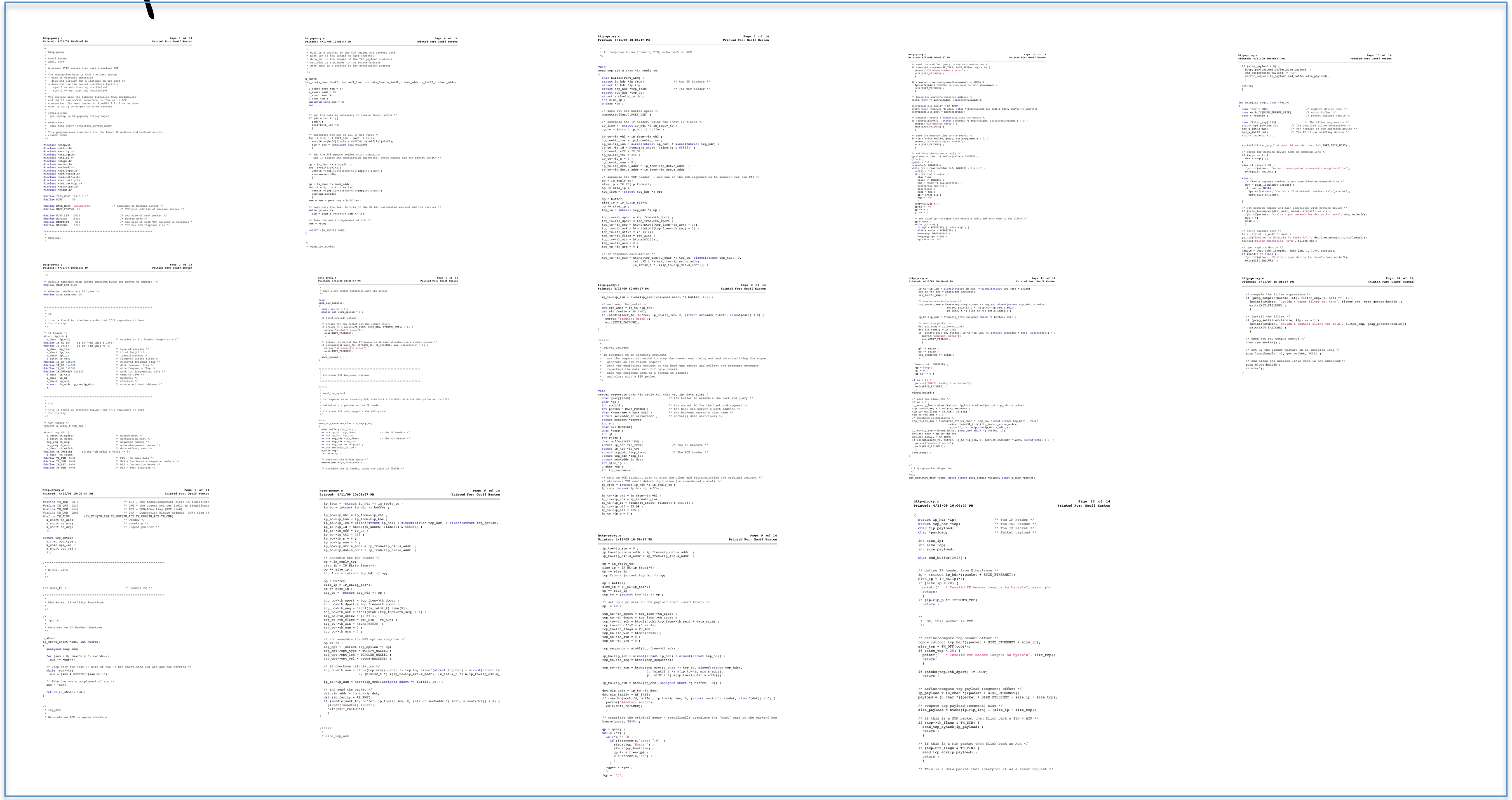
How can a user application send raw IP packets?

raw sockets is useful here

```
/* open a raw socket interface for output */
sock_fd = socket(PF_INET, SOCK_RAW, IPPROTO_TCP))  ;

/* inform the kernel the IP header is already attached via a socket option */
setsockopt(sock_fd, IPPROTO_IP, IP_HDRINCL, &on, sizeof(on)) ;

/* write a raw IP packet into the socket */
sendto(sock_fd, buffer, ip_to->ip_len, 0, (struct sockaddr *) &dst, sizeof(dst)
```

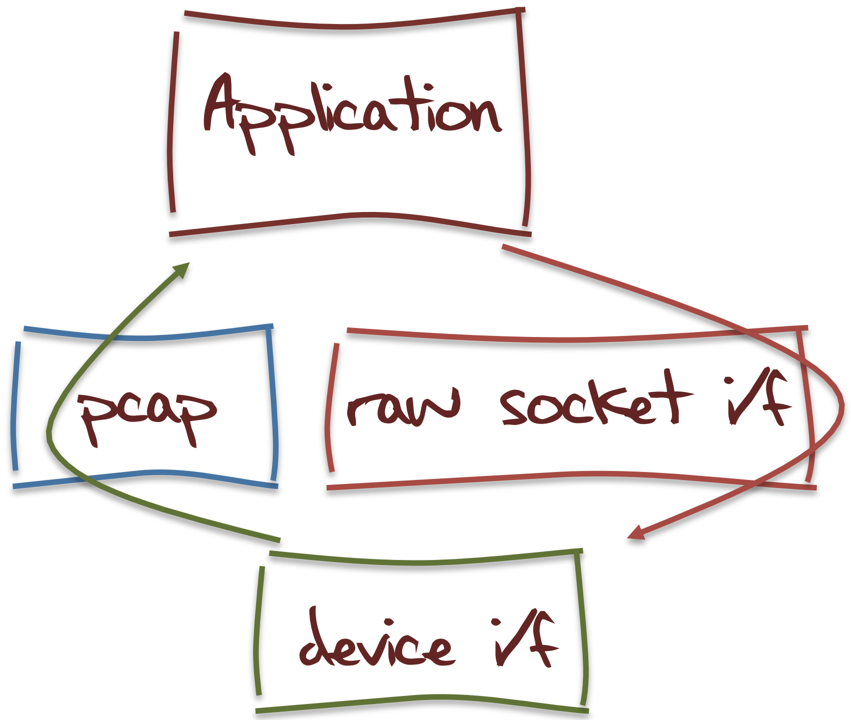
Can this be coded?

And complete the picture with crc calculator, and IP packet decoder and stateless transaction routines



Can this be coded?

Yes! A user space implementation of a stateless HTTP TCP server that avoids kernel TCP processing



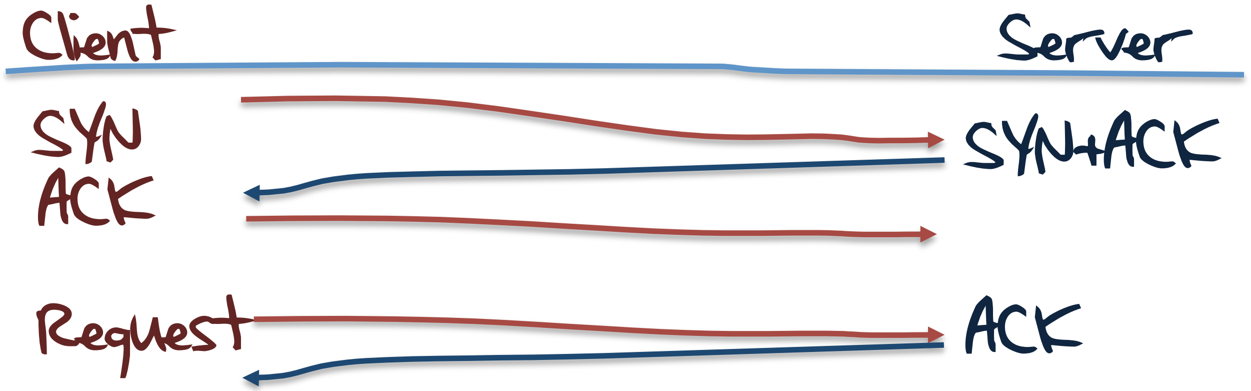
A Stateless HTTP Proxy in the wild!

```
IP client.55371 > server.http:  S 926841556:926841556(0) win 65535 <mss 1460>
IP server.http > client.55371:  S 1256778255:1256778255(0) ack 926841557 win 65535 <mss 1220>
IP client.55371 > server.http:  . ack 1 win 65535
IP client.55371 > server.http:  P 1:246(245) ack 1 win 65535
IP server.http > client.55371:  . ack 246 win 65535

IP server.56447 > backend.http: S 3055447836:3055447836(0) win 65535 <mss 1460>
IP backend.http > server.56447: S 2086147938:2086147938(0) ack 3055447837 win 65535 <mss 1460>
IP server.56447 > backend.http: . ack 1 win 8326
IP server.56447 > backend.http: P 1:248(247) ack 1 win 8326>
IP backend.http > server.56447: P 1:468(467) ack 248 win 8326

IP server.http > client.55371:  . 1:468(467) ack 246 win 65535

IP backend.http > server.56447: F 468:468(0) ack 248 win 8326
IP server.56447 > backend.http: . ack 469 win 8326
IP server.56447 > backend.http: F 248:248(0) ack 469 win 8326
IP backend.http > server.56447: . ack 249 win 8325

IP server.http > client.55371:  F 468:468(0) ack 246 win 65535
IP client.55371 > server.http:  . ack 468 win 65535
IP client.55371 > server.http:  . ack 469 win 65535
IP client.55371 > server.http:  F 246:246(0) ack 469 win 65535
IP server.http > client.55371:  . ack 247 win 65535
```

A Stateless HTTP Proxy

1. Sync and Request

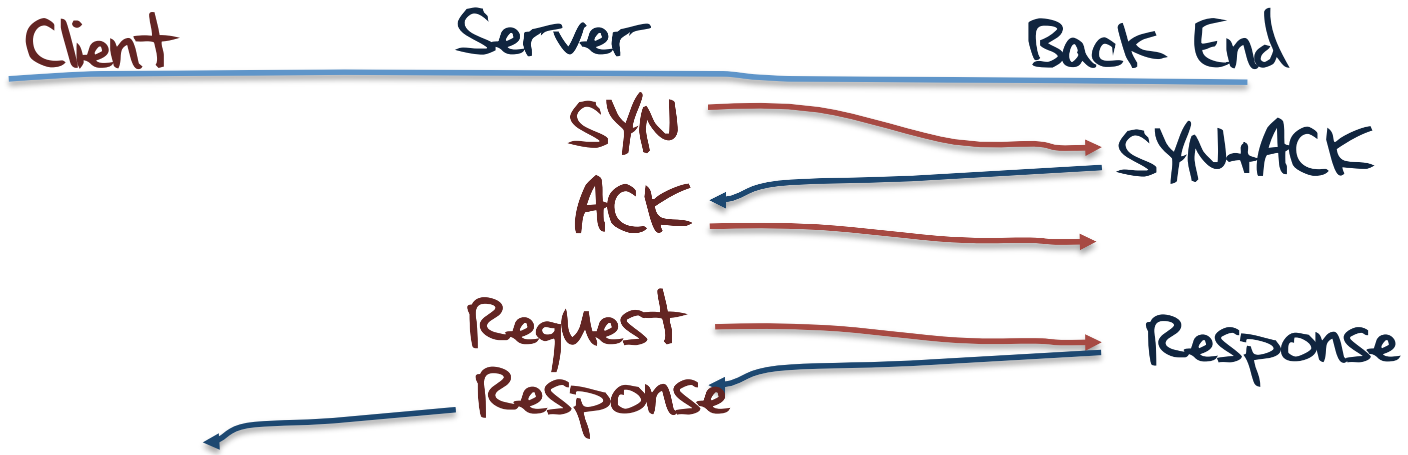
```
IP client.55371 > server.http:  S 926841556:926841556(0) win 65535 <mss 1460>
IP server.http > client.55371:  S 1256778255:1256778255(0) ack 926841557 win 65535 <mss 1220>
IP client.55371 > server.http:  . ack 1 win 65535

IP client.55371 > server.http:  P 1:246(245) ack 1 win 65535
IP server.http > client.55371:  . ack 246 win 65535
```



A Stateless HTTP Proxy

2. Backend Request and Response

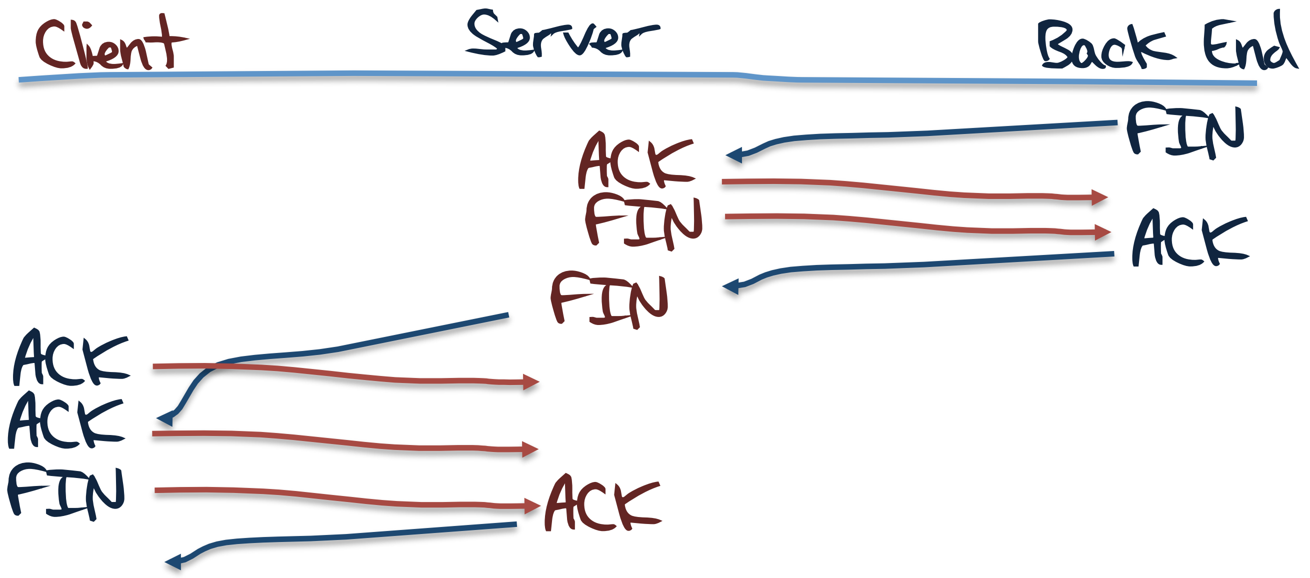
```
IP server.56447 > backend.http: S 3055447836:3055447836(0) win 65535 <mss 1460>
IP backend.http > server.56447: S 2086147938:2086147938(0) ack 3055447837 win 65535 <mss 1460>
IP server.56447 > backend.http: . ack 1 win 8326
IP server.56447 > backend.http: P 1:248(247) ack 1 win 8326>
IP backend.http > server.56447: P 1:468(467) ack 248 win 8326

IP server.http > client.55371:  . 1:468(467) ack 246 win 65535
```



A Stateless HTTP Proxy

3, shutdown

```
IP backend.http > server.56447: F 468:468(0) ack 248 win 8326
IP server.56447 > backend.http: . ack 469 win 8326
IP server.56447 > backend.http: F 248:248(0) ack 469 win 8326
IP backend.http > server.56447: . ack 249 win 8325

IP server.http > client.55371:  F 468:468(0) ack 246 win 65535
IP client.55371 > server.http:  . ack 468 win 65535
IP client.55371 > server.http:  . ack 469 win 65535
IP client.55371 > server.http:  F 246:246(0) ack 469 win 65535
IP server.http > client.55371:  . ack 247 win 65535
```

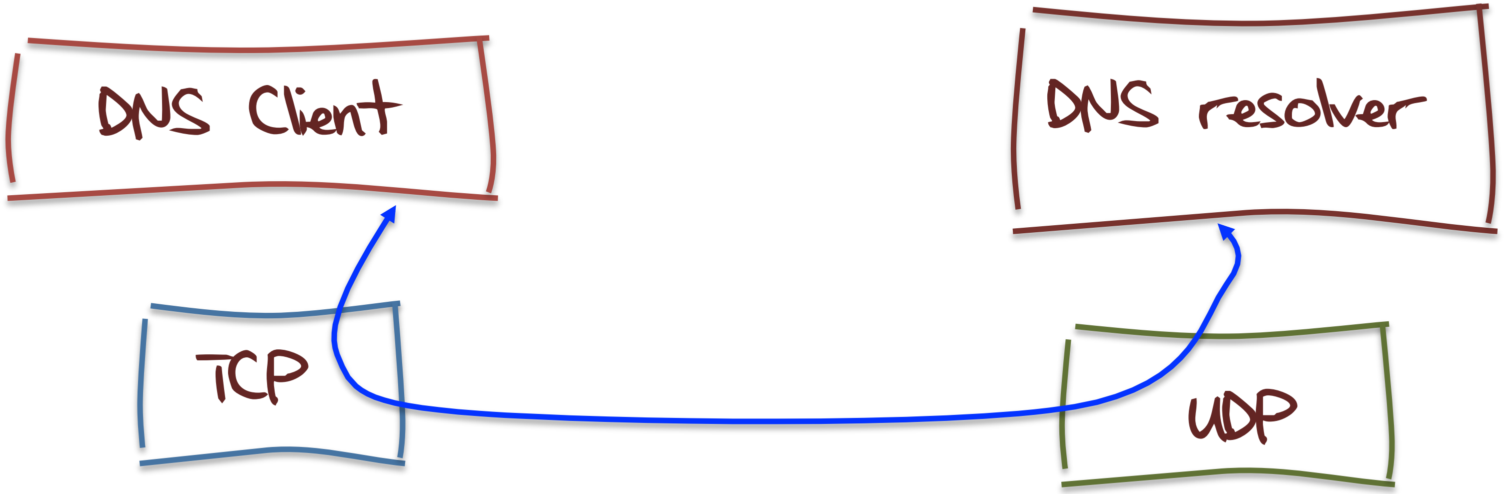


So far so good..

But can we make this bad idea a little worse?

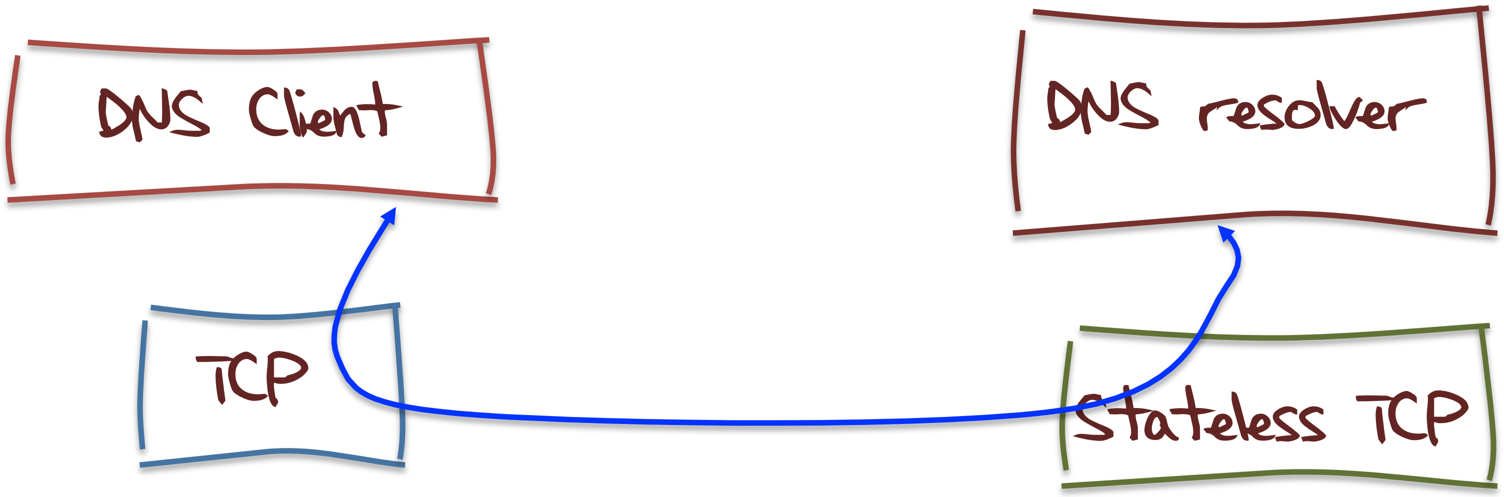
What about the DNS?

Can we use the same approach to create a hybrid model of a TCP DNS client speaking to a UDP DNS resolver?



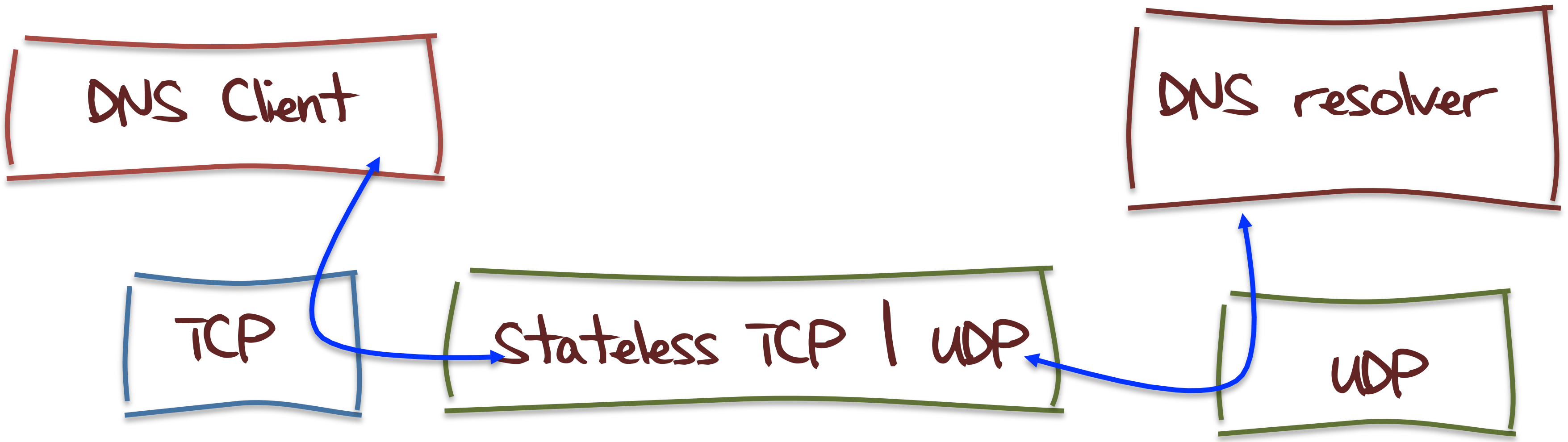
So far so good..

Or, more realistically, can we use the same approach to create a hybrid model of a TCP DNS client speaking to a stateless TCP DNS resolver?



DNS and Stateless TCP

To test if this approach could work I used a prototype config of a stateless TCP facing the client, and a UDP referral to a DNS resolver as the back end



It Worked!

```
$ dig +tcp @server rand.apnic.net in any

client.55998 > server.domain: S, cksum 0x9159 (correct), 2201103970:2201103970(0) win 65535 <mss 1460>
server.domain > client.55998: S, cksum 0x82b9 (correct), 1256795928:1256795928(0) ack 2201103971 win 65535 <mss 1220>
client.55998 > server.domain: ., cksum 0x9986 (correct), 1:1(0) ack 1 win 65535
client.55998 > server.domain: P, cksum 0x41b2 (correct), 1:35(34) ack 1 win 6553530304+ ANY? rand.apnic.net. (32)
server.domain > client.55998: ., cksum 0x9964 (correct), 1:1(0) ack 35 win 65535
server.54054 > backend.domain: 30304+ ANY? rand.apnic.net. (32)
backend.domain > server.54054: 30304* q: ANY? rand.apnic.net. 6/0/2 rand.apnic.net. SOA mirin.apnic.net. research.apnic.net.
                    2009051502 3600 900 3600000 3600, rand.apnic.net. NS mirin.apnic.net., rand.apnic.net. NS sec3.apnic.net.,
                    rand.apnic.net. MX kombu.apnic.net. 100, rand.apnic.net. MX karashi.apnic.net. 200, rand.apnic.net. MX
                    fennel.apnic.net. 300 ar: sec3.apnic.net. A sec3.apnic.net, sec3.apnic.net. AAAA sec3.apnic.net (229)
server.domain > client.55998: ., cksum 0x421a (correct), 1:232(231) ack 35 win 6553530304* q: ANY? rand.apnic.net. 6/0/2
                    rand.apnic.net. SOA mirin.apnic.net. research.apnic.net.
                    2009051502 3600 900 3600000 3600, rand.apnic.net. NS mirin.apnic.net., rand.apnic.net. NS sec3.apnic.net.,
                    rand.apnic.net. MX kombu.apnic.net. 100, rand.apnic.net. MX karashi.apnic.net. 200, rand.apnic.net. MX
                    fennel.apnic.net. 300 ar: sec3.apnic.net. A sec3.apnic.net, sec3.apnic.net. AAAA sec3.apnic.net (229)
server.domain > client.55998: F, cksum 0x987c (correct), 232:232(0) ack 35 win 65535
client.55998 > server.domain: ., cksum 0x987d (correct), 35:35(0) ack 232 win 65535
client.55998 > server.domain: ., cksum 0x987c (correct), 35:35(0) ack 233 win 65535
client.55998 > server.domain: F, cksum 0x987b (correct), 35:35(0) ack 233 win 65535
server.domain > client.55998: ., cksum 0x987c (correct), 232:232(0) ack 36 win 65535
```

It Worked!

1, TCP handshake

```
dig +tcp @server rand.apnic.net in any

client.55998 > server.domain: S, cksum 0x9159 (correct), 2201103970:2201103970(0) win 65535 <mss 1460>
server.domain > client.55998: S, cksum 0x82b9 (correct), 1256795928:1256795928(0) ack 2201103971 win 65535 <mss 1220>
client.55998 > server.domain: ., cksum 0x9986 (correct), 1:1(0) ack 1 win 65535
client.55998 > server.domain: P, cksum 0x41b2 (correct), 1:35(34) ack 1 win 6553530304+ ANY? rand.apnic.net. (32)
server.domain > client.55998: ., cksum 0x9964 (correct), 1:1(0) ack 35 win 65535
server.54054 > backend.domain: 30304+ ANY? rand.apnic.net. (32)
backend.domain > server.54054: 30304* q: ANY? rand.apnic.net. 6/0/2 rand.apnic.net. SOA mirin.apnic.net. research.apnic.net.
                    2009051502 3600 900 3600000 3600, rand.apnic.net. NS mirin.apnic.net., rand.apnic.net. NS sec3.apnic.net.,
                    rand.apnic.net. MX kombu.apnic.net. 100, rand.apnic.net. MX karashi.apnic.net. 200, rand.apnic.net. MX
                    fennel.apnic.net. 300 ar: sec3.apnic.net. A sec3.apnic.net, sec3.apnic.net. AAAA sec3.apnic.net (229)
server.domain > client.55998: ., cksum 0x421a (correct), 1:232(231) ack 35 win 6553530304* q: ANY? rand.apnic.net. 6/0/2
                    rand.apnic.net. SOA mirin.apnic.net. research.apnic.net.
                    2009051502 3600 900 3600000 3600, rand.apnic.net. NS mirin.apnic.net., rand.apnic.net. NS sec3.apnic.net.,
                    rand.apnic.net. MX kombu.apnic.net. 100, rand.apnic.net. MX karashi.apnic.net. 200, rand.apnic.net. MX
                    fennel.apnic.net. 300 ar: sec3.apnic.net. A sec3.apnic.net, sec3.apnic.net. AAAA sec3.apnic.net (229)
server.domain > client.55998: F, cksum 0x987c (correct), 232:232(0) ack 35 win 65535
client.55998 > server.domain: ., cksum 0x987d (correct), 35:35(0) ack 232 win 65535
client.55998 > server.domain: ., cksum 0x987c (correct), 35:35(0) ack 233 win 65535
client.55998 > server.domain: F, cksum 0x987b (correct), 35:35(0) ack 233 win 65535
server.domain > client.55998: ., cksum 0x987c (correct), 232:232(0) ack 36 win 65535
```

It Worked!

2. TCP request and referral to UDP DNS backend

```
dig +tcp @server rand.apnic.net in any

client.55998 > server.domain: S, cksum 0x9159 (correct), 2201103970:2201103970(0) win 65535 <mss 1460>
server.domain > client.55998: S, cksum 0x82b9 (correct), 1256795928:1256795928(0) ack 2201103971 win 65535 <mss 1220>
client.55998 > server.domain: ., cksum 0x9986 (correct), 1:1(0) ack 1 win 65535
client.55998 > server.domain: P, cksum 0x41b2 (correct), 1:35(34) ack 1 win 6553530304+ ANY? rand.apnic.net. (32)
server.domain > client.55998: ., cksum 0x9964 (correct), 1:1(0) ack 35 win 65535
server.54054 > backend.domain: 30304+ ANY? rand.apnic.net. (32)
backend.domain > server.54054: 30304* q: ANY? rand.apnic.net. 6/0/2 rand.apnic.net. SOA mirin.apnic.net. research.apnic.net.
                    2009051502 3600 900 3600000 3600, rand.apnic.net. NS mirin.apnic.net., rand.apnic.net. NS sec3.apnic.net.,
                    rand.apnic.net. MX kombu.apnic.net. 100, rand.apnic.net. MX karashi.apnic.net. 200, rand.apnic.net. MX
                    fennel.apnic.net. 300 ar: sec3.apnic.net. A sec3.apnic.net, sec3.apnic.net. AAAA sec3.apnic.net (229)
server.domain > client.55998: ., cksum 0x421a (correct), 1:232(231) ack 35 win 6553530304* q: ANY? rand.apnic.net. 6/0/2
                    rand.apnic.net. SOA mirin.apnic.net. research.apnic.net.
                    2009051502 3600 900 3600000 3600, rand.apnic.net. NS mirin.apnic.net., rand.apnic.net. NS sec3.apnic.net.,
                    rand.apnic.net. MX kombu.apnic.net. 100, rand.apnic.net. MX karashi.apnic.net. 200, rand.apnic.net. MX
                    fennel.apnic.net. 300 ar: sec3.apnic.net. A sec3.apnic.net, sec3.apnic.net. AAAA sec3.apnic.net (229)
server.domain > client.55998: F, cksum 0x987c (correct), 232:232(0) ack 35 win 65535
client.55998 > server.domain: ., cksum 0x987d (correct), 35:35(0) ack 232 win 65535
client.55998 > server.domain: ., cksum 0x987c (correct), 35:35(0) ack 233 win 65535
client.55998 > server.domain: F, cksum 0x987b (correct), 35:35(0) ack 233 win 65535
server.domain > client.55998: ., cksum 0x987c (correct), 232:232(0) ack 36 win 65535
```

It Worked!

3. TCP response to client

```
dig +tcp @server rand.apnic.net in any

client.55998 > server.domain: S, cksum 0x9159 (correct), 2201103970:2201103970(0) win 65535 <mss 1460>
server.domain > client.55998: S, cksum 0x82b9 (correct), 1256795928:1256795928(0) ack 2201103971 win 65535 <mss 1220>
client.55998 > server.domain: ., cksum 0x9986 (correct), 1:1(0) ack 1 win 65535
client.55998 > server.domain: P, cksum 0x41b2 (correct), 1:35(34) ack 1 win 6553530304+ ANY? rand.apnic.net. (32)
server.domain > client.55998: ., cksum 0x9964 (correct), 1:1(0) ack 35 win 65535
server.54054 > backend.domain: 30304+ ANY? rand.apnic.net. (32)
backend.domain > server.54054: 30304* q: ANY? rand.apnic.net. 6/0/2 rand.apnic.net. SOA mirin.apnic.net. research.apnic.net.
                    2009051502 3600 900 3600000 3600, rand.apnic.net. NS mirin.apnic.net., rand.apnic.net. NS sec3.apnic.net.,
                    rand.apnic.net. MX kombu.apnic.net. 100, rand.apnic.net. MX karashi.apnic.net. 200, rand.apnic.net. MX
                    fennel.apnic.net. 300 ar: sec3.apnic.net. A sec3.apnic.net, sec3.apnic.net. AAAA sec3.apnic.net (229)
server.domain > client.55998: ., cksum 0x421a (correct), 1:232(231) ack 35 win 6553530304* q: ANY? rand.apnic.net. 6/0/2
                    rand.apnic.net. SOA mirin.apnic.net. research.apnic.net.
                    2009051502 3600 900 3600000 3600, rand.apnic.net. NS mirin.apnic.net., rand.apnic.net. NS sec3.apnic.net.,
                    rand.apnic.net. MX kombu.apnic.net. 100, rand.apnic.net. MX karashi.apnic.net. 200, rand.apnic.net. MX
                    fennel.apnic.net. 300 ar: sec3.apnic.net. A sec3.apnic.net, sec3.apnic.net. AAAA sec3.apnic.net (229)
server.domain > client.55998: F, cksum 0x987c (correct), 232:232(0) ack 35 win 65535
client.55998 > server.domain: ., cksum 0x987d (correct), 35:35(0) ack 232 win 65535
client.55998 > server.domain: ., cksum 0x987c (correct), 35:35(0) ack 233 win 65535
client.55998 > server.domain: F, cksum 0x987b (correct), 35:35(0) ack 233 win 65535
server.domain > client.55998: ., cksum 0x987c (correct), 232:232(0) ack 36 win 65535
```

It Worked!

4. FIN close

```
dig +tcp @server rand.apnic.net in any

client.55998 > server.domain: S, cksum 0x9159 (correct), 2201103970:2201103970(0) win 65535 <mss 1460>
server.domain > client.55998: S, cksum 0x82b9 (correct), 1256795928:1256795928(0) ack 2201103971 win 65535 <mss 1220>
client.55998 > server.domain: ., cksum 0x9986 (correct), 1:1(0) ack 1 win 65535
client.55998 > server.domain: P, cksum 0x41b2 (correct), 1:35(34) ack 1 win 6553530304+ ANY? rand.apnic.net. (32)
server.domain > client.55998: ., cksum 0x9964 (correct), 1:1(0) ack 35 win 65535
server.54054 > backend.domain: 30304+ ANY? rand.apnic.net. (32)
backend.domain > server.54054: 30304* q: ANY? rand.apnic.net. 6/0/2 rand.apnic.net. SOA mirin.apnic.net. research.apnic.net.
                 2009051502 3600 900 3600000 3600, rand.apnic.net. NS mirin.apnic.net., rand.apnic.net. NS sec3.apnic.net.,
                 rand.apnic.net. MX kombu.apnic.net. 100, rand.apnic.net. MX karashi.apnic.net. 200, rand.apnic.net. MX
                 fennel.apnic.net. 300 ar: sec3.apnic.net. A sec3.apnic.net, sec3.apnic.net. AAAA sec3.apnic.net (229)
server.domain > client.55998: ., cksum 0x421a (correct), 1:232(231) ack 35 win 6553530304* q: ANY? rand.apnic.net. 6/0/2
                 rand.apnic.net. SOA mirin.apnic.net. research.apnic.net.
                 2009051502 3600 900 3600000 3600, rand.apnic.net. NS mirin.apnic.net., rand.apnic.net. NS sec3.apnic.net.,
                 rand.apnic.net. MX kombu.apnic.net. 100, rand.apnic.net. MX karashi.apnic.net. 200, rand.apnic.net. MX
                 fennel.apnic.net. 300 ar: sec3.apnic.net. A sec3.apnic.net, sec3.apnic.net. AAAA sec3.apnic.net (229)
server.domain > client.55998: F, cksum 0x987c (correct), 232:232(0) ack 35 win 65535
client.55998 > server.domain: ., cksum 0x987d (correct), 35:35(0) ack 232 win 65535
client.55998 > server.domain: ., cksum 0x987c (correct), 35:35(0) ack 233 win 65535
client.55998 > server.domain: F, cksum 0x987b (correct), 35:35(0) ack 233 win 65535
server.domain > client.55998: ., cksum 0x987c (correct), 232:232(0) ack 36 win 65535
```

What's it good for?

- It can get around the TCP state overhead in busy servers
- And by shifting up the segmentation / reassembly function from the IP layer to TCP it can circumvent firewall and difficult IPv6 PMTU/UDP issues when dealing with large responses in transaction applications
  - for an example here think DNSSEC + DNS!

But ...

Its just like UDP in almost every respect:

no reliability, no flow control, and absolutely no polite manners whatsoever!

And its still a Bad Idea!

Acknowledgements

This Bad Idea was cooked up with George Michaelson

The code to perform pseudo TCP in user space used ingredient ideas from:

“TCP/IP Illustrated, Volume 1,” Stevens

“Unix network Programming,” Stevens, Fenner & Rudoff

tcpdump and the libpcap library, Van Jacobsen et al,

http://www.tcpdump.org

The FreeBSD recipe used here for the Stateless HTTP and DNS proxies can be found at: http://www.potaroo.net/tools/useless