Playing with QUIC

Build the QUIC client and server

A sample server and client implementation are provided in Chromium. To use these you should first have checked out the Chromium source, and then build the binaries:

```
ninja -C out/Debug quic_server quic_client
```

Prepe test data from www.example.org

Download a copy of www.example.org, which we will serve locally using the quic_server binary:

```
mkdir /tmp/quic-data
cd /tmp/quic-data
wget -p --save-headers https://www.example.org
```

Manually edit index.html and adjust the headers:

- **Remove (if it exists):** "Transfer-Encoding: chunked"
- **Remove (if it exists):** "Alternate-Protocol: ..."
- **Add:** X-Original-Url: https://www.example.org/

Generate certificates

In order to run the server, you will need a valid certificate, and a private key in pkcs8 format. If you don't have one, there are scripts you can use to generate them:

```
cd net/tools/quic/certs
./generate-certs.sh
cd -
```

In addition to the server's certificate and public key, this script will also generate a CA certificate (`net/tools/quic/certs/out/2048-sha256-root.pem`) which you will need to add to your OS's root certificate store in order for it to be trusted during certificate validate. For doing this on linux, please see these instructions.

Run the QUIC server and client

Run the quic_server:

```
./out/Debug/quic_server \
  --quic_response_cache_dir=/tmp/quic-data/www.example.org \
```

```
  --certificate_file=net/tools/quic/certs/out/leaf_cert.pem \
  --key_file=net/tools/quic/certs/out/leaf_cert.pkcs8
```

And you should be able to successfully request the file over QUIC using quic_client:

```
./out/Debug/quic_client --host=127.0.0.1 --
port=6121 https://www.example.org/
```

Note that if you let the server's port default to 6121, you must specify the client port because it defaults to 80.
Moreover, if your local machine has multiple loopback addresses (as it would if using both IPv4 and IPv6), you have to pick a specific address.
It remains to be determined whether the latter shortcoming is a bug.

note: both the client and server are meant mainly for integration testing: neither is performant at scale!

To test the same download using chrome,

```
chrome \
  --user-data-dir=/tmp/chrome-profile \
  --no-proxy-server \
  --enable-quic \
  --origin-to-force-quic-on=www.example.org:443 \
  --host-resolver-rules='MAP www.example.org:443
127.0.0.1:6121' \
  https://www.example.org
```

Troubleshooting

If you run into troubles, try running the server or client with --v=1. It will increase the logging verbosity and the additional logs will often help expose the underlying problem.

评论

您没有权限添加评论。