LEARN

NGINX HTTP/3—Configurations, Troubleshooting, and Best Practices

HTTP/3, the latest version of the Hypertext Transfer Protocol, improves upon its prec by utilizing QUIC, a transport layer network protocol based on UDP. This shift addres issues like connection reliability, latency, and security inherent in HTTP/2 and HTTP/

NGINX, a popular web server and reverse proxy, offers HTTP/3 support through its *ngx_http_v3_module*. Although an experimental feature at the time of writing, this int a significant step in web technology advancement. It aims to combine NGINX's perfo and versatility with HTTP/3's advanced features.

This article explores NGINX HTTP/3 configurations, best practices, and troubleshoot

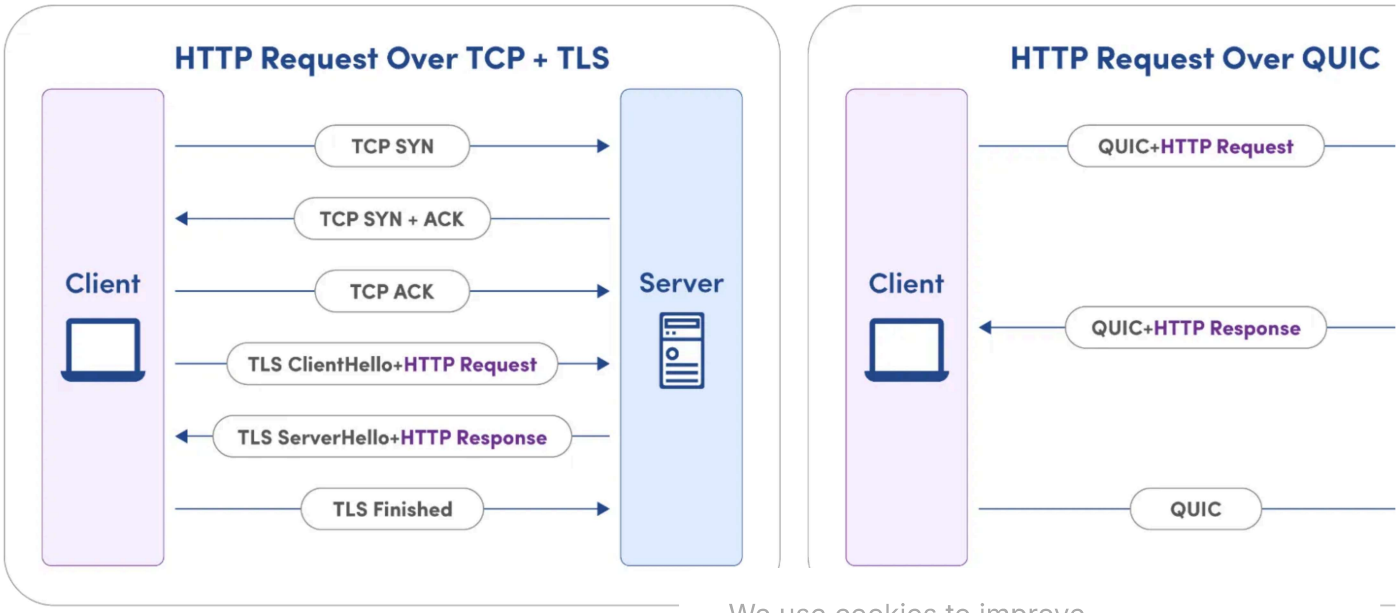
Summary of key NGINX HTTP/3 conce

Concept	Description
HTTP/3	HTTP/3 is the third and latest version of the Hypertext Transfer Protocol. It aims t latency by enabling multiplexed connections and improving security and reliability predecessors.
QUIC	Quick UDP Internet Connections(QUIC) is the transport layer protocol that HTTP/3 based on UDP instead of TCP.
NGINX	TNGINX is an open-source web server that functions as a reverse proxy, load bal proxy, and HTTP cache. It is known for its rich feature set, simple configuration, a resource consumption.

NGINX HTTP/3 introduction

The history of HTTP began as a simple protocol for transferring hypertext document Internet, evolving through versions HTTP/1.0 to HTTP/2. However, these versions fac challenges such as head-of-line blocking (explained in Chapter 1) and latency due to reliance on TCP.

HTTP/3, the latest version, aims to overcome these limitations by adopting QUIC (Qu Internet Connections), which uses UDP to reduce connection establishment time and congestion control. It allows for multiplexed streams without head-of-line blocking, s new standard for web communications.



HTTP/3 works

NGINX is a web server created by Igor Sysoev in 2004 to solve the C10k problem—t challenge of handling ten thousand concurrent connections on a single server. It wa to use an asynchronous, event-driven approach to managing requests, distinguishin the process or thread-based connection handling in traditional web servers like Apa

Initially developed as a web server to serve static content efficiently, NGINX has sinc for everything from load balancer/reverse proxy to API gateway and content caching

NGINX's supports HTTP/3 through the ngx_http_v3_module.

Deploying HTTP/3 on NGINX

Check NGINX version

First, check your NGINX version. Support for QUIC and HTTP/3 has only been availa 1.25.0.

```
nginx -v
```

You should see output similar to this:

```
nginx version: nginx/1.25.0
built with OpenSSL 3.0.2 15 Mar 2022
TLS SNI support enabled
configure arguments: --with-cc-opt='-g -O2 -ffile-prefix-map=/build/n
--lock-path=/var/lock/nginx.lock --pid-path=/run/nginx.pid
--with-http_v3_module --http-client-body-temp-path=/var/lib/nginx/bod
```

Optionally, you can also try:

```
nginx -V 2>&1 | grep -o with-http_v3_module
```

If you see with-http_v3_module in the output, your NGINX supports HTTP/3.

Build NGINX with required modules

Download a version compatible with HTTP/3 and the QUIC patch from the official re
needed. 1.25.3 is the latest version as of the time of writing this. You can replace the
the latest version currently available for you.

```
wget http://nginx.org/download/nginx-1.25.3.tar.gz
tar -zxvf nginx-1.25.3.tar.gz
cd nginx-1.25.3
```

Compile NGINX with HTTP/3 Support

A basic configuration command might look like this:

```
./configure \
    --prefix=/etc/nginx \
    --with-http_ssl_module \
    --with-http_v2_module \
    --with-http_v3_module \
    --with-openssl=$QUICHE_DIR/deps/boringssl \
    --with-quiche=$QUICHE_DIR
```

Adjust the *--prefix* and other options according to your requirements.

When configuring nginx, it is possible to enable QUIC and HTTP/3 using the --with-http_v3_module configuration parameter.

To build nginx, it is recommended that an SSL library that provides QUIC support, su BoringSSL, LibreSSL, or QuicTLS, be used. Otherwise, the OpenSSL compatibility la used, which does not support early data.

Use the following command to configure nginx with BoringSSL:

```
./configure
    --with-debug
    --with-http_v3_module
    --with-cc-opt="-I../boringssl/include"
    --with-ld-opt="-L../boringssl/build/ssl
                   -L../boringssl/build/crypto"
```

Alternatively, NGINX can be configured with QuicTLS:

```
./configure
    --with-debug
    --with-http_v3_module
    --with-cc-opt="-I../quictls/build/include"
    --with-ld-opt="-L../quictls/build/lib"
```

Alternatively, NGINX can be configured w

```
./configure
    --with-debug
    --with-http_v3_module
    --with-cc-opt="-I../libressl/build/include"
    --with-ld-opt="-L../libressl/build/lib"
```

After configuring, compile and install NGINX:

```
make
sudo make install
```

After installation, you can verify HTTP/3 support by starting NGINX and checking its configuration or using tools that detect HTTP/3 support.

```
/path/to/nginx -V
```

Confirm --with-http_v3_module is contained in the output.

Enable HTTP/3 in NGINX server block

An NGINX server block is a configuration unit within the NGINX web server that cont specific directives to define how to respond to requests for a particular domain or su Server blocks enable NGINX to process requests differently based on various reque parameters.

Below, we give a basic example of server block configuration for HTTP/3.

```
server {
    # Listen on standard HTTP/2 and HTTPS ports
    listen 443 ssl http2;
    listen [::]:443 ssl http2;

    # Listen on UDP for QUIC+HTTP/3
    listen 443 quic reuseport;
    listen [::]:443 quic reusepor
```

```
    # Specify the key and certificate files
    ssl_certificate /path/to/your/fullchain.pem;
    ssl_certificate_key /path/to/your/privkey.pem;

    # Specify the protocols including QUIC and HTTP/3
    ssl_protocols TLSv1.3;
    ssl_prefer_server_ciphers on;

    # Add Alt-Svc header to advertise HTTP/3 support to clients
    add_header Alt-Svc 'h3-23=":443"'; # Note: 'h3-23' denotes the dr

    # Server root, index files, server name and other configurations
    root /var/www/html;
    index index.html index.htm;
    server_name example.com;

    ...
}
```

QUIC-specific security settings

Configuring security for NGINX is crucial for data integrity and trust. To optimize for HTTP/3, use OpenSSL 1.1.1 or later for TLS 1.3 support. You should also specify ssl_p and ssl_ciphers tailored for optimal performance and compatibility with HTTP/3.

```
# SSL optimizations for QUIC
ssl_protocols TLSv1.3; # Ensure only TLS 1.3 is used
ssl_ciphers [your-preferred-ciphers-for-TLS1.3];
```

Advanced Configuratio

For more advanced setups, especially in high-traffic environments, consider tuning a parameters like ssl_session_cache, ssl_session_tickets, and ssl_session_timeout to o TLS handshake times and overall server performance. We give an example below.

```
server {
    listen 443 ssl http2 default_server;
    listen [::]:443 ssl http2 default_server;
    listen 443 quic reuseport default_server;
    listen [::]:443 quic reuseport default_server;

    ssl_certificate /path/to/signed_cert_plus_intermediates;
    ssl_certificate_key /path/to/private_key;
    ssl_session_timeout 1d;
    ssl_session_cache shared:MozSSL:10m;  # about 40000 sessions
    ssl_session_tickets off;

    # curl https://ssl-config.mozilla.org/ffdhe2048.txt > /path/to/dh
    ssl_dhparam /path/to/dhparam;

    # intermediate configuration
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SH
    ssl_prefer_server_ciphers off;

    # Enable QUIC and HTTP/3
    ssl_quic on;
    ssl_early_data on;
    add_header Alt-Svc 'h3=":$server_port"; ma=86400';

    # HSTS (ngx_http_headers_module is required) (63072000 seconds)
    add_header Strict-Transport-Security "max-age=63072000" always;

    # OCSP stapling
    ssl_stapling on;
    ssl_stapling_verify on;
```

```
    # verify chain of trust of OCSP response using Root CA and Interm
    ssl_trusted_certificate /path/to/root_CA_cert_plus_intermediates;

    # replace with the IP address of your resolver
    resolver 127.0.0.1;
}
```

Remember, the above configurations are a starting point. Monitoring and adjusting configurations based on performance data is crucial. Identifying bottlenecks and unc how different settings impact user experience across various global locations are cri successful deployment.

Troubleshooting tips for NGINX HTTP/

If you run into configuration issues, consider the following.

Debug SSL/TLS misconfiguration

Tools like OpenSSL can help you inspect SSL certificates and test the SSL handshak

```
openssl x509 -in /path/to/cert.pem -text -noout
```

This command displays the details of your certificate to verify its validity.

```
openssl s_client -connect yourserver.com:443 -alpn h3-29
```

Replace h3-29 with the appropriate HTTP/3 version supported by your server. This t
SSL handshake and shows if HTTP/3 negotiation occurs.

Debug QUIC protocol issues

If you're developing or debugging applications using QUIC, quic-go (a QUIC implem
Go) provides logging capabilities to trace QUIC protocol operations. You can also us
Wireshark to capture and analyze QUIC traffic. For a quick command-line check, you
tcpdump.

```
sudo tcpdump -i any -U -w quic_traffic.pcap 'udp port 443'
```

This captures UDP traffic on port 443 and saves it to a file for later analysis. Remem
decrypting QUIC traffic requires the SSL/TLS keys because of QUIC's built-in encryp

Review NGINX logs

Always check NGINX's error logs for clues on what might be going wrong. They can
you directly to the source of the problem.

```
tail -f /var/log/nginx/error.log
```

Remember, when modifying NGINX configurations or firewall rules, make incrementa
and test each step to isolate the cause of any issues.

Another consideration is the relatively new state of HTTP/3 support across clients ar
networks; while major browsers support it, not all do by default. Some enterprise net
not yet be optimized for UDP traffic assoc

Best practices for NGINX HTTP/3 configurations

Best practices for deploying NGINX with HTTP/3 are given below.

Include TLS 1.3 in configurations

Include enabling TLS 1.3, as QUIC requires. Make sure these lines appear in your cor
shown in the example above:

```
# Enable QUIC and HTTP/3
    http3 on;
    ssl_early_data on;
    add_header Alt-Svc 'h3=":$server_port"; ma=86400';
```

Additional configuration options can be found in the official documentation for NGIN

Check firewall and network configuration

Ensure your firewall allows UDP traffic on port 443. You can use iptables or ufw on L
check and modify firewall rules.

```
sudo iptables -L -v -n
```

Example with ufw.

```
sudo ufw status
```

Look for firewall rules allowing or blocking UDP traffic on port 443, and adjust accor
Ensure there's an allow rule for 443/udp.

Implement phased rollo

Test your HTTP/3 setup in a staging environment before rolling it to production. This to identify and resolve any compatibility or performance issues.

Additionally, the impact on SEO and user analytics should be considered, as the shif may affect metrics collection and analysis. To mitigate potential issues, employ a pha and monitor performance and user feedback closely.

It's also important to stay abreast of NGINX news and updates, as improvements an related to HTTP/3 are continually being released.

Performance optimization

Performance optimization for HTTP/3 on NGINX focuses on minimizing latency an throughput. This can involve tuning the underlying UDP settings for QUIC, adju reuse parameters, and optimizing TLS configuration to reduce handshake tim changing settings based on real-world usage patterns are crucial, as the e can vary significantly depending on specific traffic loads, content type conditions.

In this context, the Catchpoint platform ca deep visibility into every aspect of the Int

application performance monitoring, but not for your app stack—for your Internet St over 40 out-of-the-box monitors and the world's largest global observability networ test and monitor all your Internet Stack components, including backbone, wireless, l nodes, and cloud nodes.

Catchpoint also provides real user monitoring to track users' actions while engaging website or application. It provides accurate data and metrics to better understand us and insight into a system or service's performance. With Catchpoint's support, busir ensure that their transition to HTTP/3 enhances performance and security and align best user experience outcomes.

Final thoughts

NGINX support for HTTP/3 is indicative of the coming era in global Internet operatio users to take advantage of HTTP/3's benefits, including reduced latency, improved s and better handling of multiple data streams.

However, given the early stages of adoption, ensure you test your changes thorough deploying to production. Monitoring network usage will allow you to optimize your a configuration changes.

What's Next?

Introduction

Learn how HTTP/3 improves over HTTP/2 in multiplexing, connections, security, err etc., to improve web communication, performance, and user experience.

1. **gRPC HTTP/3**

   Learn how and why to use gRPC and HTTP/3 to build high performance modern a that perform competitively at scale.

2. **NGINX HTTP/3**

   Learn basic and advanced NGINX HTTP/3 configurations, debugging tips, how to with your firewalls, performance optim

3. **DNS over HTTPS vs. TLS**

Learn about DNS over HTTPS and DNS over TLS, how they work, performance di
PowerDNS implementation, and how to choose between the two.

4. DNS over QUIC

Learn how the Domain Name System (DNS) has evolved over time and how DNS
provides a solution with improved performance and reliability.

5. Traefik HTTP/3

Learn how HTTP/3 and Traefik work together to enhance web infrastructure. Inclu
detailed static and dynamic configurations and tutorials for Traefik HTTP/3 setup.

6. TLS 1.2 vs. 1.3

Learn the differences between TLS 1.2 and 1.3 in algorithms, version control, com
forward secrecy, and more.

7. QUIC vs. TCP

Learn how to choose between QUIC and TCP for your applications. Understand Q
TCP key differences and the metrics to monitor for performance.

Previous Chapter Next Chapter