TABLE OF CONTENTS

1. Introduction

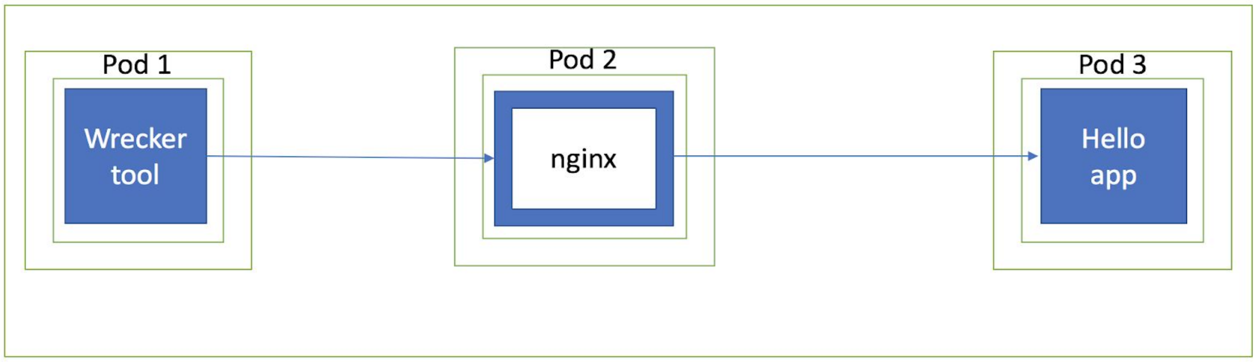
1.1 PURPOSE

Benchmarks continuously strive to improve performance standards in order to stay relevant in the market and playing important role for having better customer loyalty, SEO ranking and more. Meanwhile there are various factors affecting performance, having high performant proxy in front of webservers is one of important. By running a number of standard tests we can assess the relative performance being achieved with NGINX versus ENVOY as a proxy.

2. Test Setup

The tests were performed in 2 separate Kubernetes clusters hosted in Google Cloud Platform with same configurations. In each cluster deployed 3 containers acting as client, webserver written in Go which returns "Hello" as a response to coming client requests and reverse proxy in front of it, either nginx or envoy running. In client side we are using Wrecker load testing tool, to generate HTTP requests toward webserver.

2.1 ARCHITECTURE-NGINX



2.2 LOAD TEST STEPS

1. Create cluster in GCP:

nginx/1.13.8
a) NGINX:

project name: nginsako
cluster name: nginmesh

zone: us-east1-c
Worker Node Size: 3
Node version: 1.8.6-gke.1
Machine type: n1-standard-4 (4 vCPUs, 15 GB memory)

Total cores: 12 vCPUs
Total memory: 45.00 GB
Kernel Version: 4.4.86+

Kubernetes Cluster Description	
Parameter	Value
Zone	us-east1-c
Worker node size	3
Worker node version:	1.8.6-gke.1
Machine type:	n1-standard-4
Machine vCPUs	4
Machine Memory(Gb)	15
Kernel Version	4.4.86+
Kubernetes alpha features	Enabled

```
gcloud beta container --project "nginsako" clusters create "nginmesh" --zone "us-east1-c"
--username "admin" --cluster-version "1.8.6-gke.0" --machine-type "n1-standard-4" --image-type
"COS" --disk-size "100" --scopes
"https://www.googleapis.com/auth/compute","https://www.googleapis.com/auth/devstorage.read
_only","https://www.googleapis.com/auth/logging.write","https://www.googleapis.com/auth/mo
nitoring","https://www.googleapis.com/auth/servicecontrol","https://www.googleapis.com/auth/s
ervice.management.readonly","https://www.googleapis.com/auth/trace.append"
--enable-kubernetes-alpha --num-nodes "3" --network "default" --enable-cloud-logging
--enable-cloud-monitoring --subnetwork "default"
```

```
nginx -V
nginx version: nginx/1.13.8
built by gcc 6.3.0 20170516 (Debian 6.3.0-18)
built with OpenSSL 1.1.0f  25 May 2017
TLS SNI support enabled

configure arguments: --prefix=/etc/nginx --sbin-path=/usr/sbin/nginx
--modules-path=/usr/lib/nginx/modules --conf-path=/etc/nginx/nginx.conf
```

```
--error-log-path=/var/log/nginx/error.log --http-log-path=/var/log/nginx/access.log
--pid-path=/var/run/nginx.pid --lock-path=/var/run/nginx.lock
--http-client-body-temp-path=/var/cache/nginx/client_temp
--http-proxy-temp-path=/var/cache/nginx/proxy_temp
--http-fastcgi-temp-path=/var/cache/nginx/fastcgi_temp
--http-uwsgi-temp-path=/var/cache/nginx/uwsgi_temp
--http-scgi-temp-path=/var/cache/nginx/scgi_temp --user=nginx --group=nginx --with-compat
--with-file-aio --with-threads --with-http_addition_module --with-http_auth_request_module
--with-http_dav_module --with-http_flv_module --with-http_gunzip_module
--with-http_gzip_static_module --with-http_mp4_module --with-http_random_index_module
--with-http_realip_module --with-http_secure_link_module --with-http_slice_module
--with-http_ssl_module --with-http_stub_status_module --with-http_sub_module
--with-http_v2_module --with-mail --with-mail_ssl_module --with-stream
--with-stream_realip_module --with-stream_ssl_module --with-stream_ssl_preread_module
--with-cc-opt='-g -O2
-fdebug-prefix-map=/data/builder/debuild/nginx-1.13.8/debian/debuild-base/nginx-1.13.8=.
-specs=/usr/share/dpkg/no-pie-compile.specs -fstack-protector-strong -Wformat
-Werror=format-security -Wp,-D_FORTIFY_SOURCE=2 -fPIC'
--with-ld-opt='-specs=/usr/share/dpkg/no-pie-link.specs -Wl,-z,relro -Wl,-z,now -Wl,--as-needed
-pie'
```

b) ENVOY:

 project name: nginsako
 cluster name: envoy
 zone: us-east1-c
 Size: 3
 Node version: 1.8.6-gke.1
 Machine type: n1-standard-4 (4 vCPUs, 15 GB memory)
 Total cores: 12 vCPUs
 Total memory: 45.00 GB
Kernel Version: 4.4.86+

```
gcloud beta container --project "nginsako" clusters create "nginmesh" --zone "us-east1-c"
--username "admin" --cluster-version "1.8.6-gke.0" --machine-type "n1-standard-4" --image-type
"COS" --disk-size "100" --scopes
"https://www.googleapis.com/auth/compute","https://www.googleapis.com/auth/devstorage.read
_only","https://www.googleapis.com/auth/logging.write","https://www.googleapis.com/auth/mo
nitoring","https://www.googleapis.com/auth/servicecontrol","https://www.googleapis.com/auth/s
ervice.management.readonly","https://www.googleapis.com/auth/trace.append"
--enable-kubernetes-alpha --num-nodes "3" --network "default" --enable-cloud-logging
--enable-cloud-monitoring --subnetwork "default"
```

2. Change Kernel parameters, Port range and TCP connection reuse:

```
kubectl get nodes
gcloud compute ssh gke-nginmesh-default-pool-9c109ef3-0h99 --zone=us-east1-c
```

```
sudo vi /etc/sysctl.d/00-sysctl.conf
net.ipv4.ip_local_port_range = 1204        61000
net.ipv4.tcp_tw_reuse = 1

sudo sysctl -p /etc/sysctl.d/00-sysctl.conf
```

3. Install Wrecker, Hello-app and NGINX:

```
kubectl apply -f wrecker.yaml
kubectl apply -f hello-app.yaml
```

Proxy- Envoy or NGINX accordingly:

```
kubectl apply -f proxy/deployment.yaml
kubectl apply -f proxy/service.yaml
```

4. In NGINX install monitoring tools:

```
apt-get update && apt-get install dstat sysstat ifstat procps
```

To run from outside of container:

```
kubectl exec -it my-nginx-5d69b5ff7-6vjx9 iostat 1

kubectl exec -it my-nginx-5d69b5ff7-6vjx9 vmstat 1

kubectl exec -it my-nginx-5d69b5ff7-6vjx9 ifstat

kubectl exec -it my-nginx-5d69b5ff7-6vjx9 -- dstat --time --cpu --mem --load --net -d -p --vm
--sys 1

kubectl logs -f  my-nginx-5d69b5ff7-6vjx9
```

5. Configure NGINX to forward traffic to Hello-app:

```
cat <<EOF >/etc/nginx/conf.d/default.conf

upstream b {

        server 10.3.248.101:9080;

        keepalive 5;

}

server {
```

```
            listen  80;

            location / {

            proxy_set_header Connection "";

            proxy_http_version 1.1;

            proxy_pass http://b;

            }

}

EOF
sed -i 's/warn/notice/g' /etc/nginx/nginx.conf
```

6. Enter to the wrecker container:
kubectl exec -ti wrecker-v1-865d8dc79d-qpwks -c wrecker /bin/bash

7. Run test toward Hello-app:
  wrk -t50 –c250 –d180s http://10.3.252.9:80

8 Generating a self-signed certificate using OpenSSL:

openssl req -x509 -newkey rsa:4096 -keyout key.pem -out cert.pem -days 365 -subj /CN=localhost -passout pass:1234 -nodes

2.1 LOAD TOOL DETAILS

WRK is a modern HTTP benchmarking tool capable of generating significant load when run on a single multi-core CPU. It combines a multithreaded design with scalable event notification systems such as epoll and kqueue.

This runs a benchmark for 30 seconds, using 50 threads, and keeping 250 HTTP connections open:

```
wrk –t50 –c250 –d180s http://127.0.0.1:80/
```

Output:

```
Running 30s test @ http://127.0.0.1:80/

  12 threads and 400 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   635.91us    0.89ms  12.92ms   93.69%
    Req/Sec    56.20k     8.07k   62.00k    86.54%
  22464657 requests in 30.00s, 17.76GB read
Requests/sec: 748868.53
Transfer/sec:    606.33MB
```

IPERF3

```
iPerf3 is a tool for active measurements of the maximum achievable bandwidth on IP
networks. It supports tuning of various parameters related to timing, buffers and
protocols (TCP, UDP, SCTP with IPv4 and IPv6). For each test it reports the
bandwidth, loss, and other parameters
```

2.4 SAMPLE APPLICATION

Main.go contains the HTTP server implementation. It responds to all HTTP requests with a "Hello" response:

```
// [START all]
package main

import (
  "fmt"
  "log"
  "net/http"
  "os"
)

func main() {
  port := "8080"
  if fromEnv := os.Getenv("PORT"); fromEnv != "" {
     port = fromEnv
  }

  server := http.NewServeMux()
  server.HandleFunc("/", hello)
  log.Printf("Server listening on port %s", port)
  log.Fatal(http.ListenAndServe(":"+port, server))
}

func hello(w http.ResponseWriter, r *http.Request) {
  log.Printf("Serving request: %s", r.URL.Path)
  fmt.Fprintf(w, "Hello\n")
  }
// [END all]
```

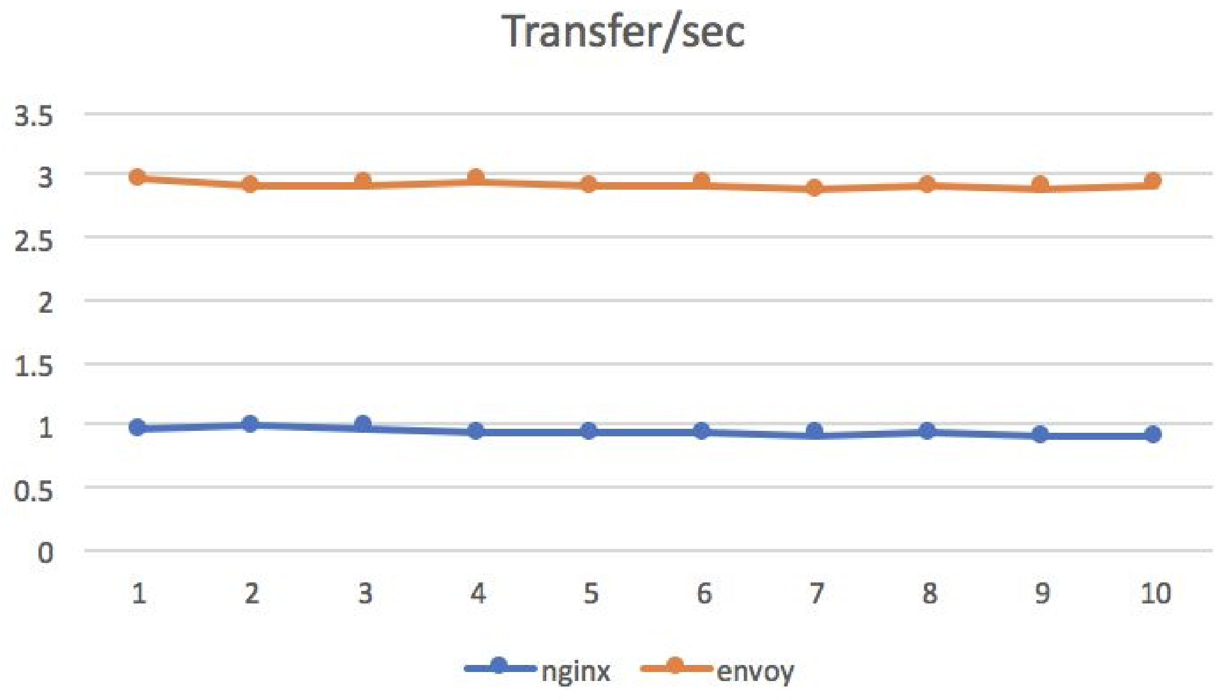
Source Code:
https://github.com/GoogleCloudPlatform/kubernetes-engine-samples/tree/master/hello-app

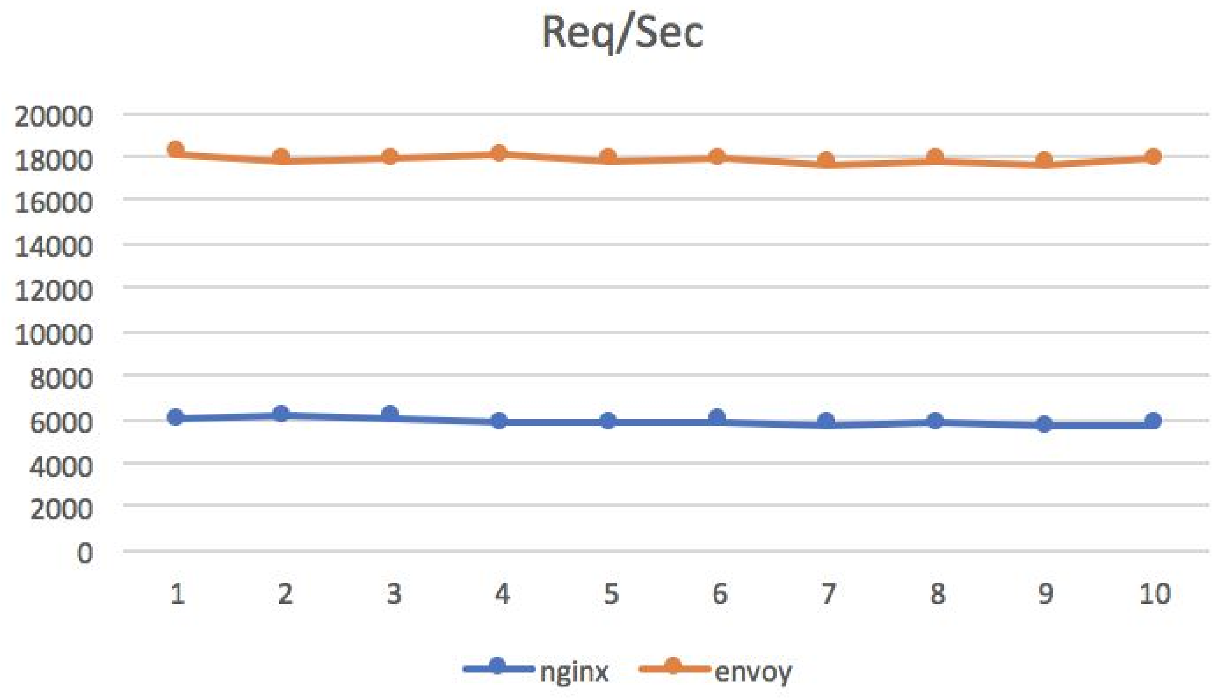
3. TEST RESULTS

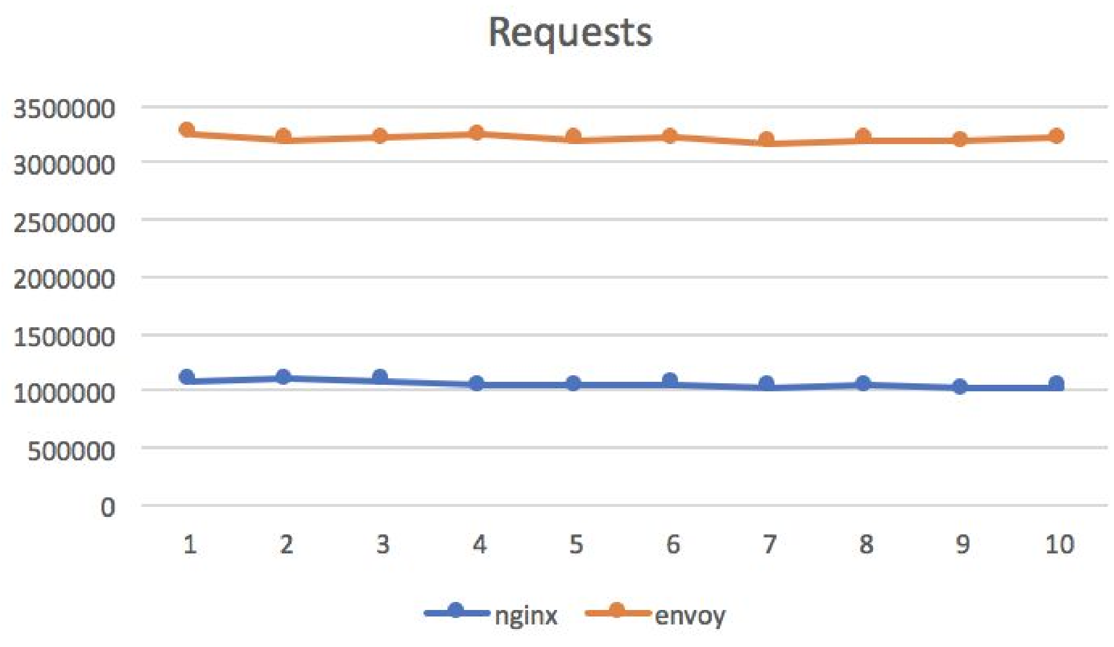
4.1 DEFAULT CONFIGURATION (HTTP)

Tool: Wrecker	
Para name	Value
thread	50
connection	250
duration	180

Hello-web APP	Requests		non-200x		req/sec		transfer/sec	
Run #	nginx	envoy	nginx	envoy	nginx	envoy	nginx	envoy
1	1081832	3260328	0	0	6010.02	18116.4	0.96	2.96
2	1108684	3203119	0	0	6159.51	17803.97	0.99	2.91
3	1093302	3218194	0	0	6073.94	17883.73	0.97	2.92
4	1053119	3246173	0	0	5850.9	18037.2	0.94	2.95
5	1047172	3206517	0	0	5817.84	17816.88	0.93	2.91
6	1061048	3210961	0	0	5895.03	17843.96	0.94	2.92
7	1037058	3176026	0	0	5761.6	17648.41	0.92	2.88
8	1046746	3205364	0	0	5815.46	17809.87	0.93	2.91
9	1022545	3179484	0	0	5680.91	17668.49	0.91	2.89
10	1026118	3212394	0	0	5700.77	17851.3	0.91	2.92



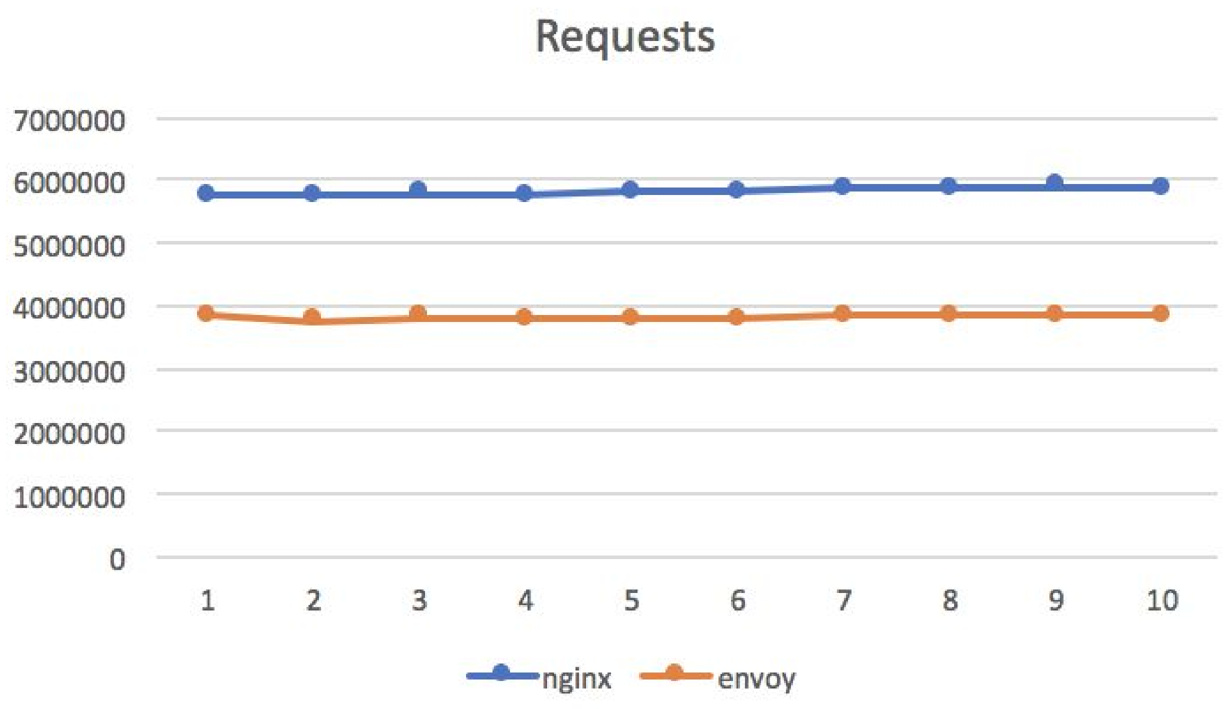

Req/Sec
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
1
2
3
4
5
6
7
8
9
10
nginx
envoy

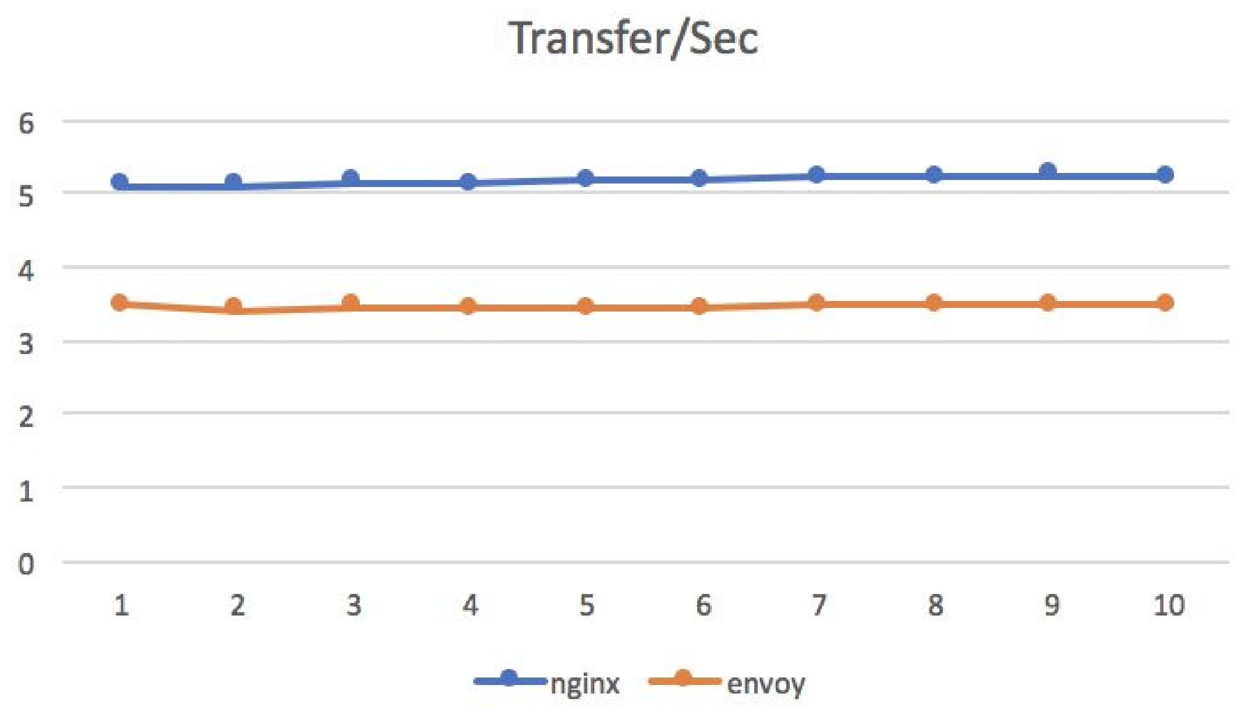

Requests
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
1
2
3
4
5
6
7
8
9
10
nginx
envoy

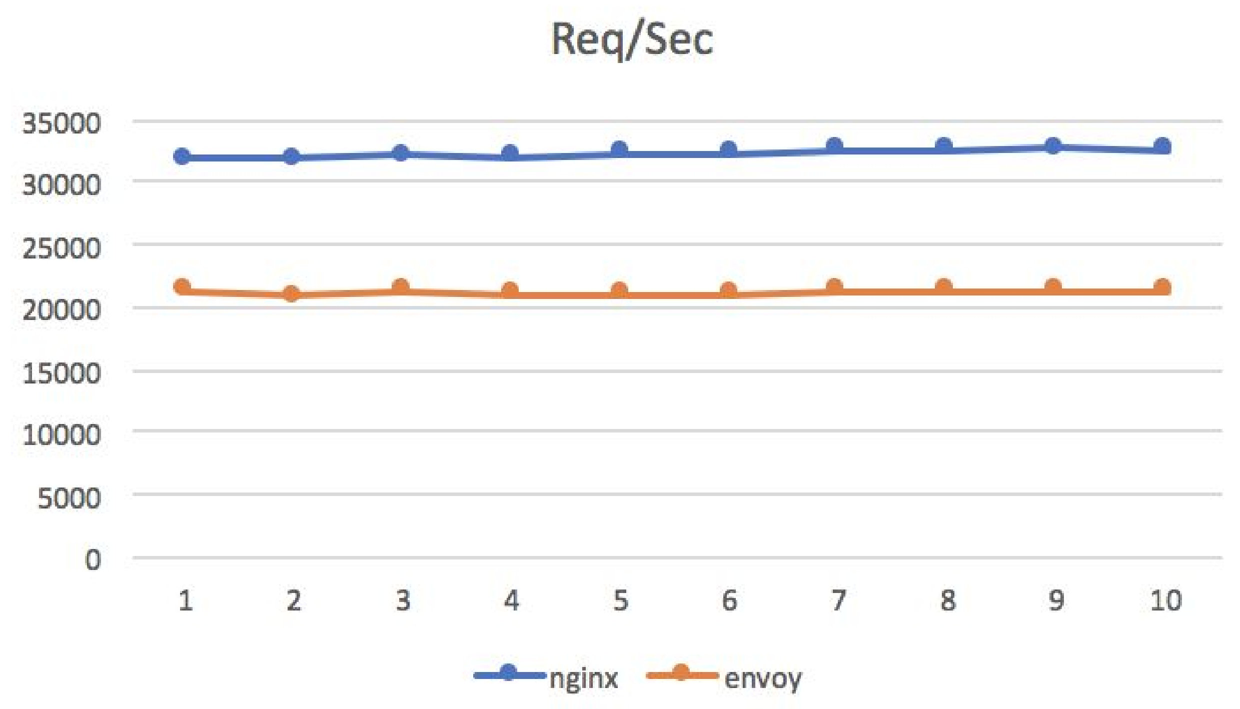
4.2 OPTIMIZED CONFIGURATION (HTTP)

Tool: Wrecker	
Para name	Value
thread	50
connection	250
duration	180

Hello-web APP	Requests		non-200x		req/sec		transfer/sec	
Run #	nginx	envoy	nginx	envoy	nginx	envoy	nginx	envoy
1	5745446	3843251	0	0	31922.53	21354.27	5.11	3.49
2	5742600	3766741	0	0	31906.45	20927.9	5.11	3.42
3	5775109	3824104	0	0	32085.76	21248.39	5.14	3.47
4	5760944	3784708	0	0	32008.12	21028.48	5.13	3.44
5	5817523	3786751	0	0	32323.15	21039.9	5.18	3.44
6	5811557	3782300	0	0	32290.37	21015.39	5.17	3.43
7	5854056	3828371	0	0	32524.81	21270.36	5.21	3.48
8	5858864	3827463	0	0	32551.2	21266.68	5.21	3.48
9	5895915	3830629	0	0	32759.18	21284.33	5.25	3.48
10	5853328	3833373	0	0	32522.39	21297.67	5.21	3.48



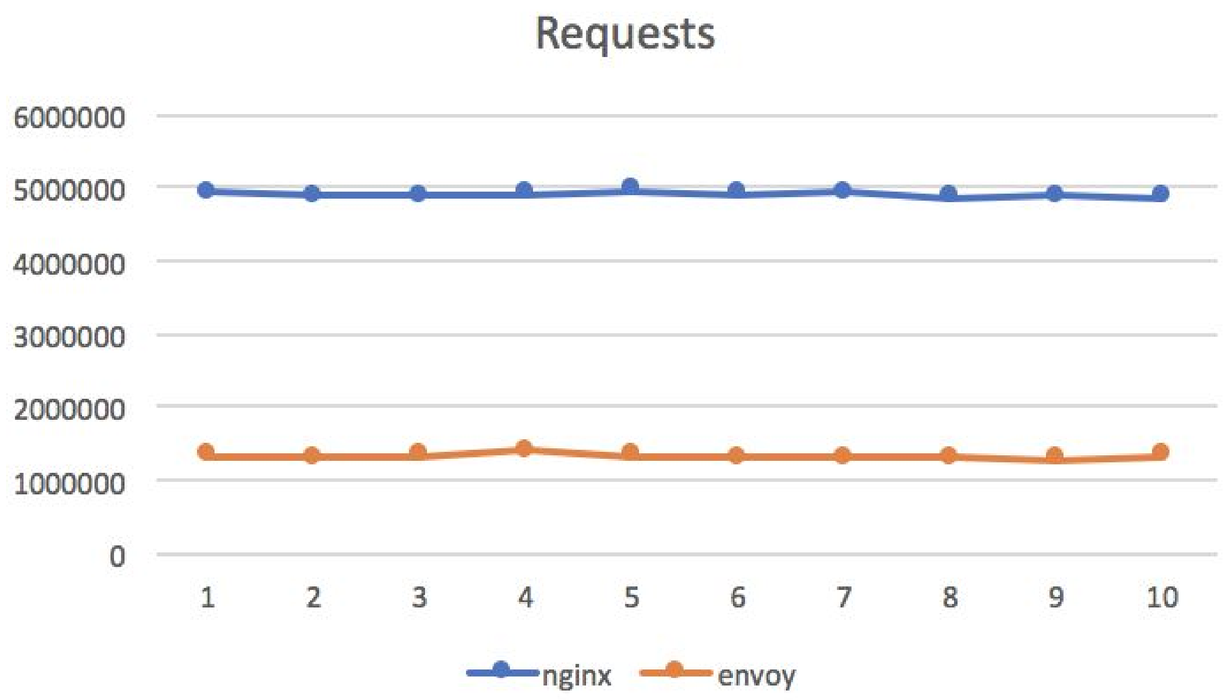

Transfer/Sec
6
5
4
3
2
1
0
1
2
3
4
5
6
7
8
9
10
nginx
envoy

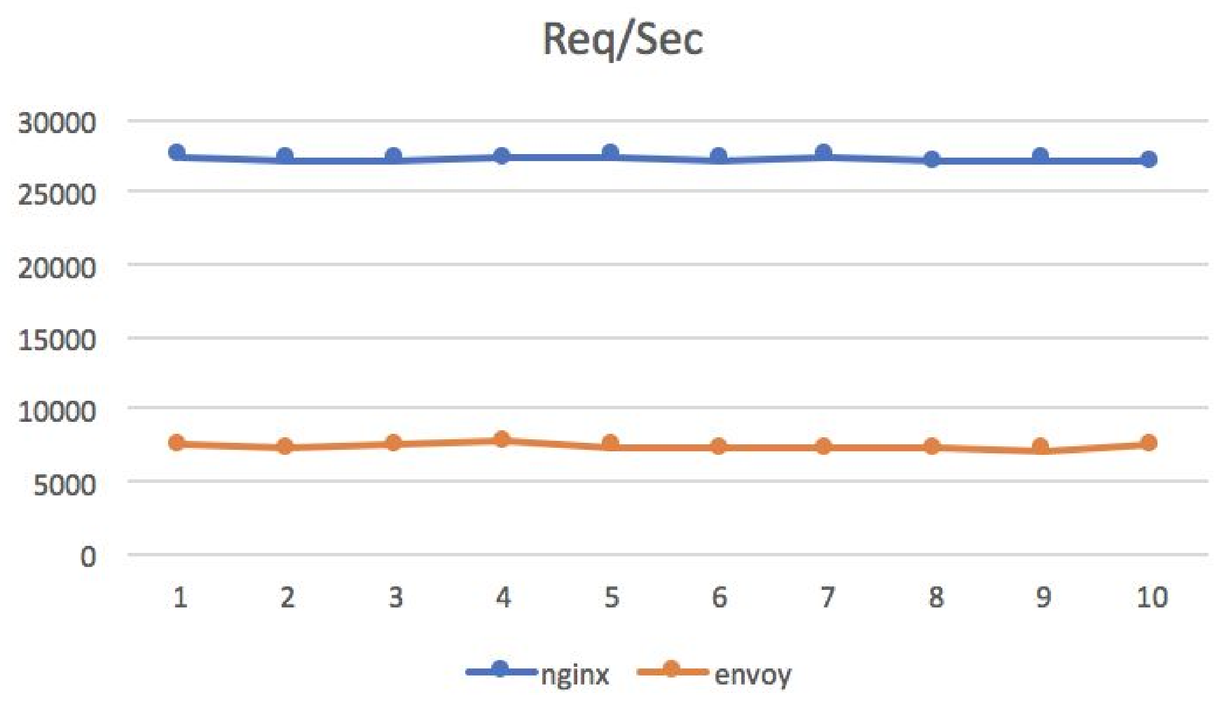

Req/Sec
35000
30000
25000
20000
15000
10000
5000
0
1
2
3
4
5
6
7
8
9
10
nginx
envoy

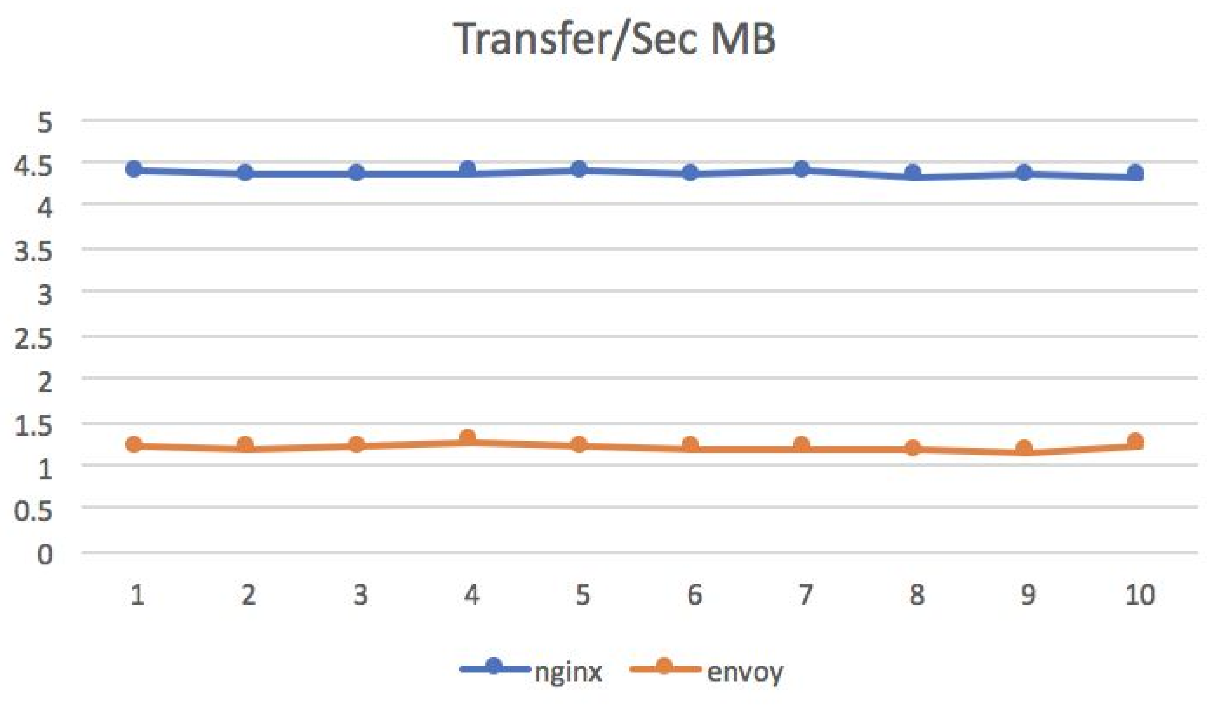
4.3 HTTPS OPTIMIZED CONFIGURATION

Hello-web APP	Requests		non-200x		req/sec		transfer/sec	
Run #	nginx	envoy	nginx	envoy	nginx	envoy	nginx	envoy
1	4930509	1344544	0	0	27408.46	7474.89	4.39	1.22
2	4883496	1306583	0	0	27139.42	7261.79	4.35	1.19
3	4889832	1344954	0	0	27184.47	7477.46	4.35	1.22
4	4906771	1400190	0	0	27276.99	7781.8	4.37	1.27
5	4945361	1334496	0	0	27483.11	7416.88	4.4	1.21
6	4902369	1305795	0	0	27236.1	7258.21	4.36	1.19
7	4924578	1312721	0	0	27377.51	7296.49	4.39	1.19
8	4866134	1300477	0	0	27052.76	7227.51	4.33	1.18
9	4889691	1280576	0	0	27176.64	7115.84	4.35	1.16
10	4862294	1347043	0	0	27033.12	7484.95	4.33	1.23

Tool: Wrecker	
Para name	Value
thread	50
connection	250
duration	180




Req/Sec
30000
25000
20000
15000
10000
5000
0
1
2
3
4
5
6
7
8
9
10
nginx
envoy


Transfer/Sec MB
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7
8
9
10
nginx
envoy

4.4 TCP BANDWIDTH(NGINX)

Run in wrecker pod:

```
iperf3 -c 10.3.243.43 -p 5000
```

Run in tcp-server pod:

```
 iperf3 -s -p 5000
```

NGINX Conf:

```
stream {
        server {
                listen     5000;
                #TCP traffic will be proxied a proxied server
                proxy_pass 10.3.243.214:9080;
    }

}
```

Output:

```
kubectl get svc
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)            AGE
go           ClusterIP   10.3.240.160   <none>        9080/TCP           5d
hello        ClusterIP   10.3.243.43    <none>        443/TCP,5000/TCP   3d
kubernetes   ClusterIP   10.3.240.1     <none>        443/TCP            27d
tcp-server   ClusterIP   10.3.243.214   <none>        9080/TCP           10m
wrecker      ClusterIP   10.3.242.37    <none>        9080/TCP           20d

iperf3 -c 10.3.243.43 -p 5000
Connecting to host 10.3.243.43, port 5000
[  4] local 10.0.1.17 port 40336 connected to 10.3.243.43 port 5000
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-1.00   sec   811 MBytes  6.81 Gbits/sec  168    868 KBytes
[  4]   1.00-2.00   sec   884 MBytes  7.42 Gbits/sec    0    980 KBytes
[  4]   2.00-3.00   sec   868 MBytes  7.28 Gbits/sec    0   1.18 MBytes
[  4]   3.00-4.00   sec   935 MBytes  7.85 Gbits/sec  172   1.08 MBytes
[  4]   4.00-5.00   sec   864 MBytes  7.25 Gbits/sec  109   1.40 MBytes
[  4]   5.00-6.00   sec   724 MBytes  6.07 Gbits/sec   15   1.43 MBytes
[  4]   6.00-7.00   sec   935 MBytes  7.84 Gbits/sec    0   1.53 MBytes
[  4]   7.00-8.00   sec   939 MBytes  7.88 Gbits/sec  145   1.11 MBytes
[  4]   8.00-9.00   sec   941 MBytes  7.90 Gbits/sec   64   1.43 MBytes
[  4]   9.00-10.00  sec   938 MBytes  7.86 Gbits/sec  290   1.12 MBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-10.00  sec  8.63 GBytes  7.41 Gbits/sec  963             sender
[  4]   0.00-10.00  sec  8.63 GBytes  7.41 Gbits/sec                  receiver
```

4.5 TCP BANDWIDTH(ENVOY)

Run in wrecker pod:

```
iperf3 -c 10.63.248.14 -p 5000
```

Run in tcp-server pod:

```
 iperf3 -s -p 5000
```

```
kubectl get svc
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
go           ClusterIP   10.63.246.249   <none>        9080/TCP   6d
hello        ClusterIP   10.63.248.14    <none>        5000/TCP   5m
kubernetes   ClusterIP   10.63.240.1     <none>        443/TCP    19d
tcp-server   ClusterIP   10.63.252.184   <none>        9080/TCP   1h
wrecker      ClusterIP   10.63.250.68    <none>        9080/TCP   19d

iperf3 -c 10.63.248.14 -p 5000
Connecting to host 10.63.248.14, port 5000
[  4] local 10.60.1.6 port 33990 connected to 10.63.248.14 port 5000
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-1.00   sec   733 MBytes  6.15 Gbits/sec  211    741 KBytes
[  4]   1.00-2.00   sec   806 MBytes  6.76 Gbits/sec  223    623 KBytes
[  4]   2.00-3.00   sec   930 MBytes  7.80 Gbits/sec  114    727 KBytes
[  4]   3.00-4.00   sec   927 MBytes  7.77 Gbits/sec  125    608 KBytes
[  4]   4.00-5.00   sec   829 MBytes  6.96 Gbits/sec   82    619 KBytes
[  4]   5.00-6.00   sec   796 MBytes  6.68 Gbits/sec   70    540 KBytes
[  4]   6.00-7.00   sec   864 MBytes  7.24 Gbits/sec   75    694 KBytes
[  4]   7.00-8.00   sec   872 MBytes  7.31 Gbits/sec  128    744 KBytes
[  4]   8.00-9.00   sec   920 MBytes  7.72 Gbits/sec  128    815 KBytes
[  4]   9.00-10.00  sec   925 MBytes  7.76 Gbits/sec  115    822 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-10.00  sec  8.40 GBytes  7.21 Gbits/sec  1271             sender
[  4]   0.00-10.00  sec  8.40 GBytes  7.21 Gbits/sec                  receiver
```

APPENDIX A: REFERENCES

Name	Description	Location
nginmesh R-0.3.0	Nginmesh project Source Code	https://github.com/nginmesh/nginmesh
WRK	Load Tool Source code	https://github.com/wg/wrk
AB	Apache Load Tool	https://httpd.apache.org/docs/2.4/programs/ab.html
Istio R-0.3.0	Istio Source Code	https://github.com/istio

APPENDIX B: NGINX Parameters Description

The following table provides definitions for NGINX parameters to be tuned:

Para name	Value	Description	Comment
worker_rlimit_nofile	100000	Change max of open files for worker process	
keepalive	5	Reuse TCP connections	Increasing helps
limit_conn_zone	$binary_remote_addr zone=conn_limit_per_ip:10m	Limit the number of connections per single IP	Increasing helps, but non 2xx increase as well
limit_req_zone	$binary_remote_addr zone=req_limit_per_ip:10m rate=5r/s	Defines the parameters for rate limiting	
access_log	off	Disable Access log	Increasing helps
worker_processes	auto	Set based on CPU Cores	Affects, 100k difference in num of requests
worker_rlimit_nofile	100000	Changes the limit on the maximum number of open files for worker process. By default OS sets to 2k	
error_log	crit	Error log level	
worker_co	4000	How many client per worker	no affect

nnection		process(64k socket system limit)	
use epoll		Serve many clients with each thread	
multi_accept	on	Accept as many connection as possible	
open_file_cache	max=200000 inactive=20s	open file descriptors, their sizes and modification times;information on existence of directories;file lookup errors, such as “file not found”, “no read permission”	
open_file_cache_valid	30s	Sets a time after which open_file_cache elements should be validated.	
open_file_cache_min_uses	2	To cache info as long as 2 requests made during 20s window	
open_file_cache_errors	on	Enable caching of file lookup errors	
sendfile	on	Syscall, execution is done inside the kernel space, replaces the combination of both read and write	
tcp_nopush	on	Activate TCP_Cork in Linux, which blocks data till packet reach MSS=MTU-40 byte of IPv4	
tcp_nodelay	on	Forces a socket to send the data in its buffer, whatever the packet size	
gzip	on	Compress the data that needs to be sent over network	
gzip_min_length	10240		
gzip_proxied	expired no-cache		

	no-store private auth		
gzip_types	text/plain text/css text/xml text/javascript application/x-javascript application/json application/xml		
gzip_disable	msie6		
reset_timedout_connection	on	Allows Server to close connection on non-responding client, will free memory	
client_body_timeout	10	request timed out, default 60 seconds	
send_timeout	2	If client stop respond, free memory, default 60 seconds	
keepalive_timeout	30	Srv will close conn after this time	
keepalive_requests	10000	number of requests client can make over keep-alive	no affect

APPENDIX C: NGINX(Version 1.13.8) Default Configuration

```
user  nginx;
worker_processes  1;

error_log  /var/log/nginx/error.log warn;
pid        /var/run/nginx.pid;

events {
  worker_connections  1024;
}

http {
  include       /etc/nginx/mime.types;
  default_type  application/octet-stream;

  log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                    '$status $body_bytes_sent "$http_referer" '
                    '"$http_user_agent" "$http_x_forwarded_for"';

  access_log  /var/log/nginx/access.log  main;
  sendfile        on;

  keepalive_timeout  65;

  include /etc/nginx/conf.d/*.conf;
}

# configuration file /etc/nginx/conf.d/default.conf:
upstream b {
  server 10.3.240.160:9080;
}

server {
  listen  80;
  location / {
     proxy_pass http://b;
  }

  error_page   500 502 503 504  /50x.html;
  location = /50x.html {
     root   /usr/share/nginx/html;
  }
}
```

APPENDIX D: NGINX(Version 1.13.8) Optimized Configuration

```
user  nginx;
worker_processes  auto;
worker_cpu_affinity auto;

error_log  /var/log/nginx/error.log warn;
pid        /var/run/nginx.pid;

events {
   worker_connections  4096;
   use epoll;
   multi_accept on;
}
worker_rlimit_nofile 100000;

http {
   include       /etc/nginx/mime.types;
   default_type  application/octet-stream;
   server_names_hash_bucket_size 64;
   log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                     '$status $body_bytes_sent "$http_referer" '
                     '"$http_user_agent" "$http_x_forwarded_for"';

   access_log  /var/log/nginx/access.log  main buffer=16k;
   include /etc/nginx/conf.d/*.conf;

}

# configuration file /etc/nginx/conf.d/default.conf:
upstream b {
   server 10.3.240.160:9080;
   keepalive 4000;
}
server {

   listen       80;
   location / {
       proxy_set_header Connection "";
       proxy_http_version 1.1;
       proxy_pass http://b;
   }
}
```

APPENDIX E: Envoy(Version 1.6.0) Default Configuration

```
{
 "listeners": [
  {
   "address": "tcp://0.0.0.0:80",
   "filters": [
    {
     "type": "read",
     "name": "http_connection_manager",
     "config": {
      "codec_type": "auto",
      "stat_prefix": "ingress_http",
      "route_config": {
       "virtual_hosts": [
        {
         "name": "service",
         "domains": ["*"],
         "routes": [
          {
           "timeout_ms": 0,
           "prefix": "/",
           "cluster": "local_service"
          }
         ]
        }
       ]
      },
      "filters": [
       {
        "type": "decoder",
        "name": "router",
        "config": {}
       }
      ]
     }
    }
   ]
  }
 ],
 "admin": {
  "access_log_path": "/tmp/envoy-access-log",
  "address": "tcp://127.0.0.1:8001"
 },
 "cluster_manager": {
  "clusters": [
   {
    "name": "local_service",
    "connect_timeout_ms": 250,
    "type": "static",
    "lb_type": "round_robin",
    "hosts": [
          {
            "url": "tcp://10.63.252.188:9080"
          }
        ]
      }
    ]
  }
}
```

APPENDIX F: Envoy(Version 1.6.0) Optimized Configuration

```
{
 "listeners": [
  {
    "address": "tcp://0.0.0.0:80",
    "filters": [
     {
       "type": "read",
       "name": "http_connection_manager",
       "config": {
         "codec_type": "auto",
         "stat_prefix": "ingress_http",
         "route_config": {
           "virtual_hosts": [
            {
              "name": "service",
              "domains": ["*"],
              "routes": [
               {
                 "timeout_ms": 2000,
                 "prefix": "/",
                 "cluster": "local_service"
               }
              ]
            }
           ]
         },
         "filters": [
           {
             "type": "decoder",
             "name": "router",
             "config": {
               "dynamic_stats": false
             }
           }
         ],
         "generate_request_id": false
       }
     }
    ]
  }
 ],
 "admin": {
   "access_log_path": "/tmp/envoy-access-log",
   "address": "tcp://127.0.0.1:8001"
 },
 "cluster_manager": {
   "clusters": [
    {
      "name": "local_service",
      "circuit_breakers": {
        "default": {
            "max_connections": 98304,
            "max_pending_requests": 98304,
            "max_requests": 98304
          }
        },
        "connect_timeout_ms": 2000,
        "type": "static",
        "lb_type": "round_robin",
        "hosts": [
          {
            "url": "tcp://10.63.246.249:9080"
          }
        ]
      }
    ]
  }
}
```

APPENDIX G: NGINX(Version 1.13.8) SSL Optimized Configuration

```
user  nginx;
worker_processes  auto;
worker_cpu_affinity auto;

error_log  /var/log/nginx/error.log warn;
pid        /var/run/nginx.pid;

events {
   worker_connections  4096;
   use epoll;
   multi_accept on;
}
worker_rlimit_nofile 100000;

http {
   include       /etc/nginx/mime.types;
   default_type  application/octet-stream;
   server_names_hash_bucket_size 64;
   log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                     '$status $body_bytes_sent "$http_referer" '
                     '"$http_user_agent" "$http_x_forwarded_for"';

   access_log  /var/log/nginx/access.log  main buffer=16k;
   include /etc/nginx/conf.d/*.conf;

}

# configuration file /etc/nginx/conf.d/default.conf:
upstream b {
   server 10.3.240.160:9080;
   keepalive 4000;
}
server {

   listen       80;
   location / {
       proxy_set_header Connection "";
       proxy_http_version 1.1;
       proxy_pass http://b;
   }
}
```

APPENDIX H: Envoy(Version 1.6.0) SSL Optimized Configuration

```
{
 "listeners": [
   {
     "address": "tcp://0.0.0.0:443",
     "ssl_context": {
       "alpn_protocols": "h2",
       "cert_chain_file": "/etc/cert.pem",
       "private_key_file": "/etc/key.pem",
       "ca_cert_file": "/etc/cert.pem"
     },
     "filters": [
       {
         "type": "read",
         "name": "http_connection_manager",
         "config": {
           "codec_type": "auto",
           "stat_prefix": "ingress_http",
           "route_config": {
             "virtual_hosts": [
               {
                 "name": "local_service",
                 "domains": ["*"],
                 "routes": [
                   {
                     "timeout_ms": 2000,
                     "prefix": "/",
                     "cluster": "local_service"
                   }
                 ]
               }
             ]
           },
           "filters": [
             {
               "type": "decoder",
               "name": "router",
               "config": {
                 "dynamic_stats": false
               }
             }
           ],
           "generate_request_id": false
         }
       }
     ]
   }
 ],
 "admin": {
   "access_log_path": "/dev/null",
   "address": "tcp://0.0.0.0:9901"
 },
 "cluster_manager": {
   "clusters": [
     {
       "name": "local_service",
       "circuit_breakers": {
         "default": {
           "max_connections": 98304,
           "max_pending_requests": 98304,
           "max_requests": 98304
         }
       },
       "connect_timeout_ms": 2000,
       "type": "static",
       "lb_type": "round_robin",
       "hosts": [
         {
           "url": "tcp://10.63.246.249:9080"
         }
       ]
     }
   ]
 }
}
```

APPENDIX I: Envoy TCP Configuration

```
{
 "listeners": [
   {
     "address": "tcp://0.0.0.0:5000",
     "filters": [
       { "type": "read", "name": "tcp_proxy",
         "config": {
           "stat_prefix": "test_tcp",
           "route_config": {
             "routes": [
               {
                 "cluster": "cluster_1"
               }
             ]
           }
         }
       }
     ]
   }
 ],
 "admin": { "access_log_path": "/dev/null", "address": "tcp://127.0.0.1:8001" },
 "statsd_udp_ip_address": "127.0.0.1:8001",
 "cluster_manager": {
   "clusters": [
     {
       "name": "cluster_1",
       "connect_timeout_ms": 5000,
       "type": "static",
       "lb_type": "round_robin",
       "hosts": [{"url": "tcp://10.63.252.184:9080"}]
     }]
 }
}
```

APPENDIX J: NGINX TCP Configuration

```
user  nginx;
worker_processes  auto;
worker_cpu_affinity auto;

error_log  /var/log/nginx/error.log warn;
pid        /var/run/nginx.pid;

events {
   worker_connections  4096;
   use epoll;
   multi_accept on;
}
worker_rlimit_nofile 100000;

stream {
       server {
               listen     5000;
               #TCP traffic will be proxied a proxied server
               proxy_pass 10.3.243.214:9080;
   }

}

http {
   include       /etc/nginx/mime.types;
   default_type  application/octet-stream;
   server_names_hash_bucket_size 64;
   log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                     '$status $body_bytes_sent "$http_referer" '
                     '"$http_user_agent" "$http_x_forwarded_for"';

   access_log  /var/log/nginx/access.log  main buffer=16k;

   include /etc/nginx/conf.d/*.conf;

}
```

APPENDIX K: NGINX Compilation Parameters

NGINX directory locations in OS:
--prefix=/etc/nginx
--sbin-path=/usr/sbin/nginx
--modules-path=/usr/lib/nginx/modules
--conf-path=/etc/nginx/nginx.conf
--error-log-path=/var/log/nginx/error.log
--http-log-path=/var/log/nginx/access.log
--pid-path=/var/run/nginx.pid
--lock-path=/var/run/nginx.lock
--http-client-body-temp-path=/var/cache/nginx/client_temp
--http-proxy-temp-path=/var/cache/nginx/proxy_temp
--http-fastcgi-temp-path=/var/cache/nginx/fastcgi_temp
--http-uwsgi-temp-path=/var/cache/nginx/uwsgi_temp
--http-scgi-temp-path=/var/cache/nginx/scgi_temp

Credentials used by worker processes:
--user=nginx
--group=nginx

Enabled additional modules:

# Enable loading compiled nginx modules to nginx-plus
--with-compat
# Enables asynchronous I/O
--with-file-aio
# Enables to use thread pools
--with-threads
# Adds text before and after a response
--with-http_addition_module
# Enables client authorization based on the result of a subrequest
--with-http_auth_request_module
# Enables file management on server via the WebDAV protocol
--with-http_dav_module
# Enables pseudo-streaming server-side support for Flash Video
--with-http_flv_module
# Unzip responses with “Content-Encoding: gzip” for clients that do not support “gzip” encoding method
--with-http_gunzip_module
# Allows sending precompressed files with the “.gz” filename extension instead of regular files.
--with-http_gzip_static_module

--with-http_mp4_module
# Processes requests ending with the slash character (‘/’) and picks a random file in a directory to serve as an index
--with-http_random_index_module
# Change the client address and optional port to those sent in the header field
--with-http_realip_module
# Check authenticity of requested links, protect resources from unauthorized access, and limit link lifetime
--with-http_secure_link_module
# Filter that splits a request into subrequests, each returning a certain range of response
--with-http_slice_module

```
# Support for HTTPS
--with-http_ssl_module
#  Provides access to basic status information
--with-http_stub_status_module
# Modifies a response by replacing one string by another
--with-http_sub_module
# Support for HTTP/2
--with-http_v2_module
# Enables mail proxy
--with-mail
# Support for a mail proxy server to work with the SSL/TLS protocol
--with-mail_ssl_module
# Enables the TCP proxy
--with-stream
# Change the client address and port to the ones sent in the PROXY protocol header
--with-stream_realip_module
# Support for a stream proxy server to work with the SSL/TLS
--with-stream_ssl_module
# Enables extracting information from the ClientHello message at the preread phase
--with-stream_ssl_preread_module
```