CDN Report:

In the CDN project we have used both the JSON as well Google Protobuf as message serializing protocol between server and client during file transfer. Below is the result of the measurement taken during the file upload process. We have used 3 cloud servers for this as Mircosoft, Google and Amazon. All these VMs are running Ubuntu linux machine. We have used document, image and pdf files for upload.

Protobuf2 Vs Json

Sr#	File name	File size kb	Source	Destination	Json Time (ms)	Protobuf3 Time (ms)
1	1.docx	32	Amazon	Microsoft	18	18
2	2.pptx	548	Amazon	Microsoft	114	149
3	3.jpg	764	Amazon	Microsoft	123	139
4	4.pdf	953	Amazon	Microsoft	141	159
5	5.jpg	5725	Amazon	Microsoft	424	474
6	6.jpg	7111	Amazon	Microsoft	606	833
7	7.pdf	7872	Amazon	Microsoft	837	926
8	8.jpg	13795	Amazon	Microsoft	565	195
9	9.pdf	31610	Amazon	Microsoft	364	566
10	10.pdf	68116	Amazon	Microsoft	599	709
11	1.docx	32	Microsoft	Google	17	18
12	2.pptx	548	Microsoft	Google	120	145
13	3.jpg	764	Microsoft	Google	125	133
14	4.pdf	953	Microsoft	Google	144	156
15	5.jpg	5725	Microsoft	Google	401	421
16	6.jpg	7111	Microsoft	Google	575	780
17	7.pdf	7872	Microsoft	Google	808	920
18	8.jpg	13795	Microsoft	Google	560	199
19	9.pdf	31610	Microsoft	Google	361	566
20	10.pdf	68116	Microsoft	Google	579	719

Conclusion:

1) The major difference between json and protobuf is that json is simple key value pair collection of object whereas the protobuf is typed one. Protobuf requires class kind of structure and it has to compiled by proto compiler.
2) The resultant experiment clearly indicated that json took less time in most of the cases.
3) The difference in time is much less when the size is less and grows when the size of file increases. However there are certain places exceptions. It might be due to reason that the network is dynamic and if the traffic increases then it might take more time.

Image scaling:

The second aim of the CDN project is to study the image compression. We are compressing the images both in static as well dynamic manner. In static, we compress the images at the time they are uploaded. For dynamic, the image scaling is done when the client requests the image. As compression we are scaling the images to two sizes: 60% , naming it as small and 75%, naming it as medium. When client requests an image file we first check the client network speed and according to it return the image. To check the client speed we are using socket connection to the server before the request comes to server via the proxy which is present at client end. Below is the chart we are using to decide the client network speed:

Slow range: if socket connection time is more than 10 ms, return the smaller size image.

Medium range: if socket connection time is more than 5 ms and less than 10 ms, return the medium size image.

Fast range: if socket connection time is less than 5 ms, return the original image.

Below is the measurement presented:

Sr#	File name	Server	Client speed	Static/Dynamic Compression	Original Size Kb	Med Scaling Kb	Small Scaling Kb
1	3.jpg	Google	10	Static	764	267	110
2	5.jpg	Google	9	Dynamic	5725	596	152
3	6.jpg	Amazon	11	Dynamic	7111	1137	372
4	8.jpg	Microsoft	7	Dynamic	13795	2823	871
5	3.jpg	Amazon	10	Static	764	267	110
6	5.jpg	Microsoft	8	Dynamic	5725	596	152
7	6.jpg	Google	12	Static	7111	1137	372
8	8.jpg	Amazon	6	Static	13795	2823	871

Image scaling after uploading the files: