Chapter 4
Experiments

In this chapter, we measure and analyze the data from the perspective of comparing essential framework, aggregative framework and selectivity content framework.

4.1 Environment

WebStone project [37] is used as our benchmark. It was developed by SGI, which gave users almost complete control over the workload characteristics, including the request sizes and mixtures. WebStone is a configurable client-server benchmark for web servers, which uses workload parameters and client processes to generate HTTPS traffic that allows a server to be stressed in a number of different ways. The process of load generation in WebStone is performed by successively requesting pages and files from the server as fast as it can process the requests. A new request is sent out to the server just after a client receives the answer of the previous request.

The experimental configuration comprises three machines listed in Table 2. These machines are interconnected directly with a cross over Ethernet line. The typical cipher suite SSL_RSA_WITH_RC4_MD5 is chosen to request both from clients and from proxy. We experimentally classify target files into range for 1kb, 2kb, 4kb, 8kb, 16kb, 32kb, 64kb, 128kb, 256kb, and 512kb in sizes. The number of simultaneous clients is given from 20 to 100 incremented by 20. The testing duration is set equally to 10 minutes, and the results presented are the average values of three rounds performed for each configuration and workload. Notably, program

performance obviously depends on hardware capacity. It is considerable that performance can be improved markedly by using more high-end processors.

Table 2. Equipments of Testbed.

WebStone	SSL Proxy Server	Web Server
PIII-1G 640MB SDRAM Intel Pro 100S RH 9.0	P4-2G 512MB DDR Intel Pro 100VE / Intel Pro 100+ RH 9.0	P-M 1.3G 256MB DDR Intel Pro 100VE RH 9.0 Appache 2.0.53

First of all, the most important metric is connection establishing rate. It derived from total established connections, divided by total test time. The other two metrics are throughput and response time. The former is total amount of bytes (body + header) transferred throughout the test, divided by the total test time, measured in bps and the latter represents the time that last byte of a response, i.e. the average response time to complete a request, from the client's standpoint.

The first result from the perspective of comparing essential framework and aggregative framework is measured and analyzed to realize the improvement degree by using resume handshake from these sets of the experiments. All detail measurement results are also given for obtained from all the counters in our experiments running on several different setups.

The second experiment is conducted with measuring the selectivity content framework. Therefore, all configurations are the same parameters used in testing except a difference cryptography suite is used. We evaluate by requesting GIF objects, and on the other side, SSL proxy server uses the specific cipher suite

SSL_RSA_WITH_NULL_MD5. The cipher suite is lacked of symmetric cryptographic RC4 compared with SSL_RSA_WITH_RC4_MD5. The experiments are evaluated according to different sizes, the results are derived as Table 3 for connection rate, Table 4 for throughput, and Table 5 for response time respectively.

4.2 Performance Evaluation

The first result of our experimental metric lies in connection establishing rate. The statistics for connection rate in different workloads based on essential framework are shown in the Table 3(a). On the other hand, the statistics based on aggregative framework and selectivity framework are shown in the Table 3(b) and Table 3(c) respectively. The performance differences mainly caused by the effect of different file sizes rather than the number of simultaneous clients. The improvement ratio is obviously observed, as illustrated from a comparison of Table 3(a), Table 3(b) with Table 3(c), it changes whenever the file size changes.

Table 3(a). Connection rate of essential framework

Essential Framework									
	1k	4k	8k	16k	32k	64k	128k	256k	512k
20	90.50	87.05	86.64	82.75	80.70	71.46	57.63	37.20	20.08
40	90.53	87.14	86.69	82.86	80.78	71.66	57.61	37.15	20.04
60	90.59	87.15	86.76	82.99	80.86	71.65	57.67	37.12	19.94
80	90.99	87.52	87.02	83.02	80.88	71.69	57.69	37.16	19.89
100	90.96	87.49	87.01	83.11	80.91	71.69	57.64	37.09	19.83

Table 3(b). Connection rate of aggregative framework

Aggregative Framework									
	1k	4k	8k	16k	32k	64k	128k	256k	512k
20	238.85	220.92	208.40	181.35	151.75	111.10	71.63	42.95	21.75
40	239.12	222.63	209.46	183.43	152.84	111.13	72.64	42.91	21.69
60	239.27	227.62	211.58	187.88	154.15	112.30	73.35	43.00	21.88
80	239.95	228.79	214.75	187.96	154.96	112.09	73.33	43.26	21.53
100	241.82	229.57	217.72	188.93	155.80	112.97	73.33	43.50	21.52

Table 3(c). Connection rate of selectivity content framework

Selectivity Content Framework									
	1k	4k	8k	16k	32k	64k	128k	256k	512k
20	239.28	224.36	213.82	190.32	162.68	121.05	81.30	43.42	22.15
40	239.69	224.96	213.93	190.35	163.69	122.04	81.34	43.37	22.09
60	240.18	229.45	215.70	193.52	164.32	124.06	82.20	44.15	21.82
80	241.40	231.18	217.73	196.91	164.77	124.85	82.26	44.25	22.03
100	245.12	235.26	219.88	198.08	165.73	124.77	82.38	43.93	21.60

Regard to the aggregative framework compared with essential framework. Given an example for 100 clients, Figure 16 summarizes that when the size of target file is small, the improvement ratio of connection rate is relative larger. The reason is explained as following: Suppose a typical environment, the total time of a request is equal to T. Let t_1 refer to the time of establishing connection and t_2 refer to the time of content transmission. Hence,

$T = t_1 + t_2$

The time of establishing connection is equal to t_1' while using resume handshake mechanism (i.e. $t_1' < t_1$). Therefore, a total time of a request is equal to T', where $T' = t_1' + t_2$. Depend upon above assumptions we can derive the improvement ratio as that:

$$\text{Improvement ratio} = \frac{t_1 + t_2}{t_1' + t_2}$$

It is result in a small size file can be coped with quickly in given testing duration and the transfer time t_2 is smaller. Consequently, improvement ratio gets a greater impact. Conversely, improvement ratio will be smaller in result of a large size file makes the transfer time t_2 longer.

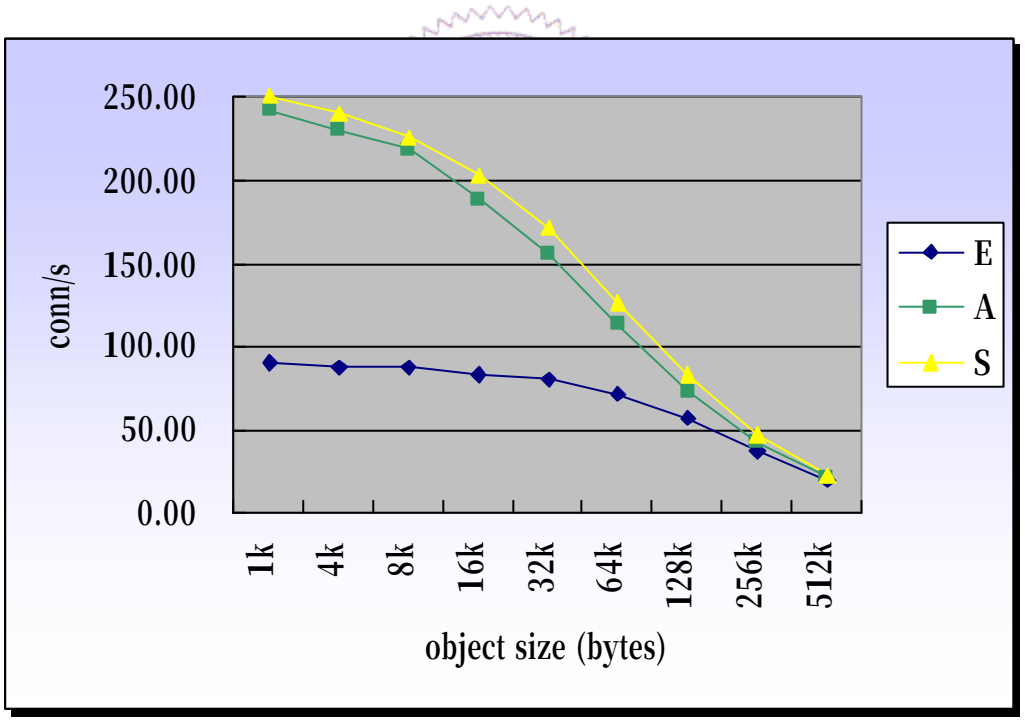

Figure 16. Comparison of connection establishing rate.

However, while comparing selectivity content framework with aggregative framework, the overall performance is impacted ideally as file size increases due to saving symmetric cryptographic computing in transferring data (include header and

payload). In other words, the improvement ratio is proportional to the size of file transferred. The improvement for aggregative framework is relative slight than that of essential framework. In fact, the improvement ratio of connection rate, regard to 1k is about 2.6, 4k is 3.6, 8k is 4.2, 16k is 6.8, 32k is 10, 64k is 12.4, and 13.8 for 128k respectively. These results are reasonable for increasing performance while only simple effort is negotiated with the Null-encryption cipher suite methodology.

We can have the same conclusion for response time and throughput: the improvement ratio of connection rate is very dependent on the file size, as shown in Table 4 and Table 5, respectively. Generally, most of objects are between 100bytes and 10kbytes in size [34]. The average size of most request pages is approximately less than 10kbytes. In [35], it showed that the average size is about 9K. Therefore, it makes sense that using aggregative framework can have remarkable improvement over essential framework. On the other hand, selectivity content framework has a positive impact of transferring the non-text representations such as image or audio files.

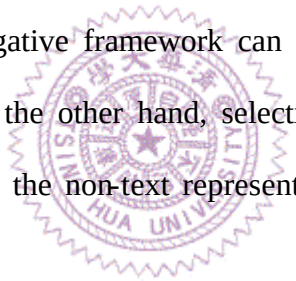

Table 4(a). Throughput of essential framework.

Essential Framework									
	1k	4k	8k	16k	32k	64k	128k	256k	512k
20	0.92	3.15	5.88	11.03	21.34	37.62	60.56	79.14	83.08
40	0.95	3.28	5.93	11.18	21.48	37.77	60.54	79.11	83.02
60	0.97	3.29	5.99	11.32	21.63	37.75	60.58	79.07	82.95
80	1.13	3.56	6.23	11.34	21.69	37.79	60.59	79.12	82.91
100	1.09	3.54	6.20	11.35	21.71	37.80	60.56	79.01	82.87

Table 4(b). Throughput of aggregative framework.

Aggregative Framework									
	1k	4k	8k	16k	32k	64k	128k	256k	512k
20	2.48	7.73	14.12	24.17	40.12	58.50	75.27	90.17	91.28
40	2.46	7.89	14.22	24.19	40.21	58.52	75.28	90.13	91.23
60	2.45	7.96	14.34	25.03	40.75	59.13	77.08	90.27	91.83
80	2.51	7.99	14.34	25.16	40.77	59.49	77.06	90.72	91.13
100	2.52	8.03	14.75	25.19	40.98	59.48	77.06	91.32	90.65

Table 4(c). Throughput of selectivity content framework.

Selectivity Content Framework									
	1k	4k	8k	16k	32k	64k	128k	256k	512k
20	2.42	7.87	14.50	25.38	43.02	63.72	85.42	91.14	92.95
40	2.46	7.91	14.58	25.46	43.04	64.69	85.43	91.18	91.91
60	2.51	8.05	14.63	25.81	43.45	65.29	85.36	92.68	91.55
80	2.57	7.99	14.69	26.00	43.53	65.60	85.52	92.23	91.83
100	2.59	8.26	14.91	26.41	43.82	65.78	86.87	92.23	90.29

Unfortunately, another observation from object sizes, the improvement degree has apparently collapsed rapidly. It derives from the nature of SSL that its maximum block size is 16KB. Suppose a 32 KB object is going to be transferred, the read function needs to be called twice. Similarly, it needs four read function calls to transfer a 64kb-size object. Therefore, the overhead of system calls results in slight improvement for entire system because of the more interruptions that the operating system has to handle (i.e., the system overhead increases). In our future work, we will

modify the OpenSSL library to eliminate this restriction. The improvement for sizes larger than 32KB would be significant.

In addition, as one might expect, the aggregative framework can have a shorter response time than that of essential framework. Their improvement ratios are close to each other. Also, the performance is compared under 8k file size, as shown in Figure 17. The different number of clients is the key impact factor. As a result, with larger number of client contribution, it results in heavy loading on proxy server. Therefore, it is proportional to response time. However, the response time, as clearly shown, is growing quickly while the number of clients increasing in the essential framework. On the contrary, the response time is growing relatively slow while the number of clients increasing in the aggregative framework and selectivity content framework.

Table 5(a). Response time of essential framework

Essential Framework									
	1k	4k	8k	16k	32k	64k	128k	256k	512k
20	0.223	0.230	0.235	0.245	0.251	0.284	0.348	0.530	1.013
40	0.440	0.466	0.478	0.492	0.504	0.561	0.689	1.003	2.132
60	0.636	0.699	0.702	0.712	0.743	0.855	0.929	1.512	3.195
80	0.820	0.875	0.914	0.953	0.997	1.139	1.260	2.096	4.371
100	1.021	1.146	1.169	1.203	1.214	1.413	1.518	2.689	5.418

Table 5(b). Response time of aggregative framework

Aggregative Framework									
	1k	4k	8k	16k	32k	64k	128k	256k	512k
20	0.087	0.090	0.096	0.110	0.132	0.180	0.279	0.464	0.912
40	0.163	0.172	0.197	0.219	0.262	0.364	0.552	0.926	1.861
60	0.252	0.263	0.282	0.318	0.388	0.532	0.813	1.388	2.681
80	0.339	0.358	0.379	0.415	0.536	0.738	1.012	1.814	3.701
100	0.412	0.434	0.458	0.527	0.641	0.880	1.348	2.271	4.453

Table 5(c). Response time of selectivity content framework

Selectivity Content Framework									
	1k	4k	8k	16k	32k	64k	128k	256k	512k
20	0.087	0.089	0.093	0.105	0.123	0.165	0.246	0.459	0.897
40	0.160	0.169	0.193	0.210	0.251	0.349	0.531	0.911	1.822
60	0.251	0.261	0.278	0.309	0.364	0.482	0.726	1.365	2.679
80	0.332	0.351	0.372	0.401	0.504	0.717	0.973	1.793	3.694
100	0.402	0.423	0.453	0.502	0.601	0.793	1.198	2.250	4.470

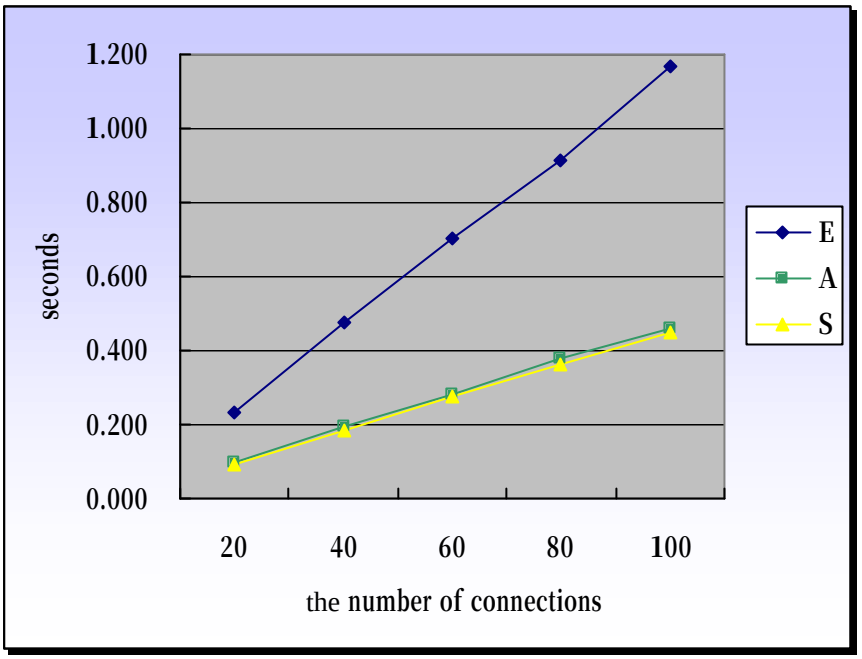

Figure 17. Comparison of the response time.