

AWS Communications – Project Report

Project Overview

Amazon EC2 servers are set up with VMs in four different locations:

- Seoul, Korea
- Dublin, Ireland
- Sao Paulo, Brazil
- Mumbai, India

The servers are aware of each other's addresses and have software to upload and download data to designated directories in each VM.

Files are separated into blocks, and such blocks are stored differently depending in what block replacement policy is used to do so.

Below are the 5 block replacement policies used across the servers.

- Random policy: when a file is upload, it chooses, for each block, a random datacenter and stores the block there.
- Tworandom policy: client chooses two datacenters at random for each block (must be two different ones).
- Local policy: store the blocks on the block store running on the same datacenter as the client (localhost).
- Localclosest policy: client stores in localhost and on the closest remaining datacenter (lowest RTT: round trip time).
- Localfarthest policy: client stores in localhost and on the farthest remaining datacenter (farthest RTT: round trip time).

Policy name	# copies of each block	Location of Block #1	Location of Block #2
random	1	random() mod N	
tworandom	2	random() mod N	random() mod N (diff than block #1)
local	1	localhost	
localclosest	2	localhost	Non-local server with min avg RTT
localfarthest	2	localhost	Non-local server with max avg RTT

Experimental Data

Policy	Experiment	RTT(Upload)			RTT(Download)		
random	Seoul (Exp. 1)	Dublin	Sao Paulo	Mumbai	Dublin	Sao Paulo	Mumbai
		267.279 61.35	298.893 0.04595	163.307 0.08675	252.106 0.07318	298.203 0.06615	163.423 0.07589
	Ireland/Seoul (Exp. 2)	Seoul	Sao Paulo	Mumbai	Dublin	Sao Paulo	Mumbai
		278.9513 75.09214	185.77025 0.04866	122.07625 0.66647	252.10075 0.09307	299.88838 0.06514	158.0712 9 0.05197
two random	Seoul (Exp. 1)	Dublin	Sao Paulo	Mumbai	Dublin	Sao Paulo	Mumbai
		256.154 39.8966	291.575 0.46018	158.017 .053434	250.456 .123881	293.374 0.68110	158.23 0.4350
	Ireland/Seoul (Exp. 2)	Seoul	Sao Paulo	Mumbai	Dublin	Sao Paulo	Mumbai
		274.869 70.0618	183.878 0.33083	119.592 0.07868	251.9791 0.083702	300.61975 0.0400598	157.8935 0.074314

Policy	Experiment	RTT(Upload)			RTT(Download)		
local	Seoul (Exp. 1)	Dublin	Sao Paulo	Mumbai	Dublin	Sao Paulo	Mumbai
		281.228 87.2092	294.749 0.07776	158.163 0.05796	241.553 0.07751	295.886 0.07258	158.069 0.048497
	Ireland/ Seoul (Exp. 2)	Seoul	Sao Paulo	Mumbai	Dublin	Sao Paulo	Mumbai
		275.962 73.06785	184.019 0.85658	119.447 0.05457	252.41113 0.2044158	295.5104 0.0322753	157.80888 0.0734311
Local closest	Seoul (Exp. 1)	Dublin	Sao Paulo	Mumbai	Dublin	Sao Paulo	Mumbai
		255.168 13.3508	294.239 0.04766	158.396 0.05130	249.798 0.07116	301.264 0.06328	158.055 0.113142
	Ireland/ Seoul (Exp. 2)	Seoul	Sao Paulo	Mumbai	Dublin	Sao Paulo	Mumbai
		273.81 67.0297	183.862 0.05970	123.686 0.025411	252.232 0.188	299.921 0.086	158.053 0.9845
Local farthest	Seoul (Exp. 1)	Dublin	Sao Paulo	Mumbai	Dublin	Sao Paulo	Mumbai
		270.02 60.895	295.95 0.0665	158.128 0.10646	249.901 0.08779	295.339 0.05531	157.836 0.09535
	Ireland/ Seoul (Exp. 2)	Seoul	Sao Paulo	Mumbai	Dublin	Sao Paulo	Mumbai
		273.918 71.8509	183.819 0.038634	123.881 0.094254	252.137 0.076	299.804 0.174	157.789 0.271

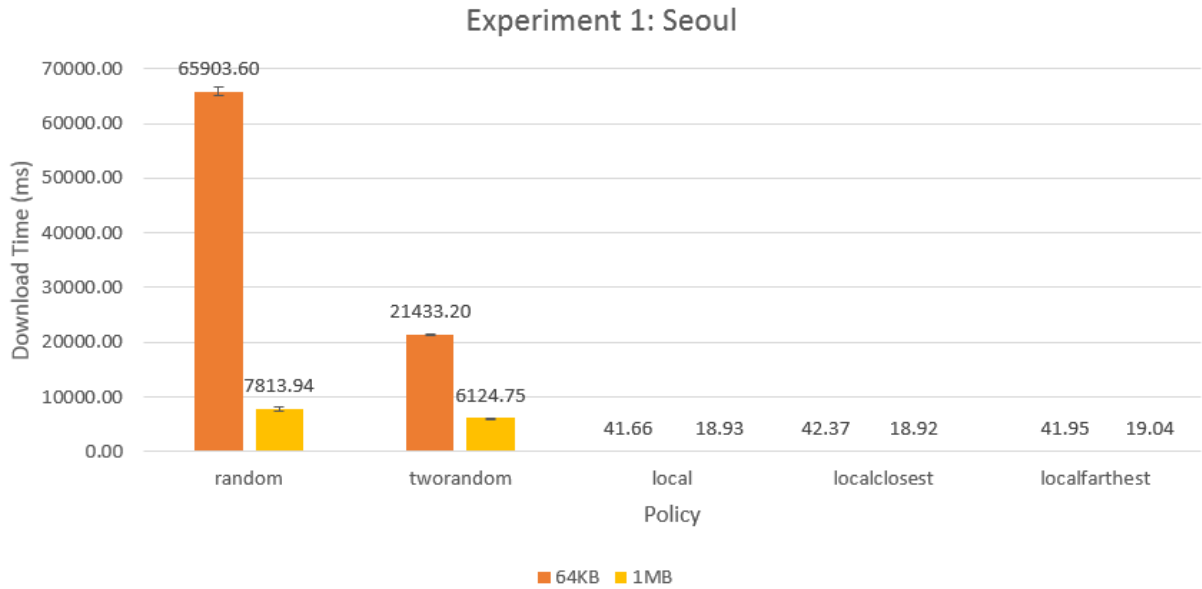


Figure 1. The above graph represents the results of three runs of Experiment 1, uploading and downloading from Seoul. The average download times of 10 1MB files using various policies are represented as bars and the standard deviations are plotted as error bars.

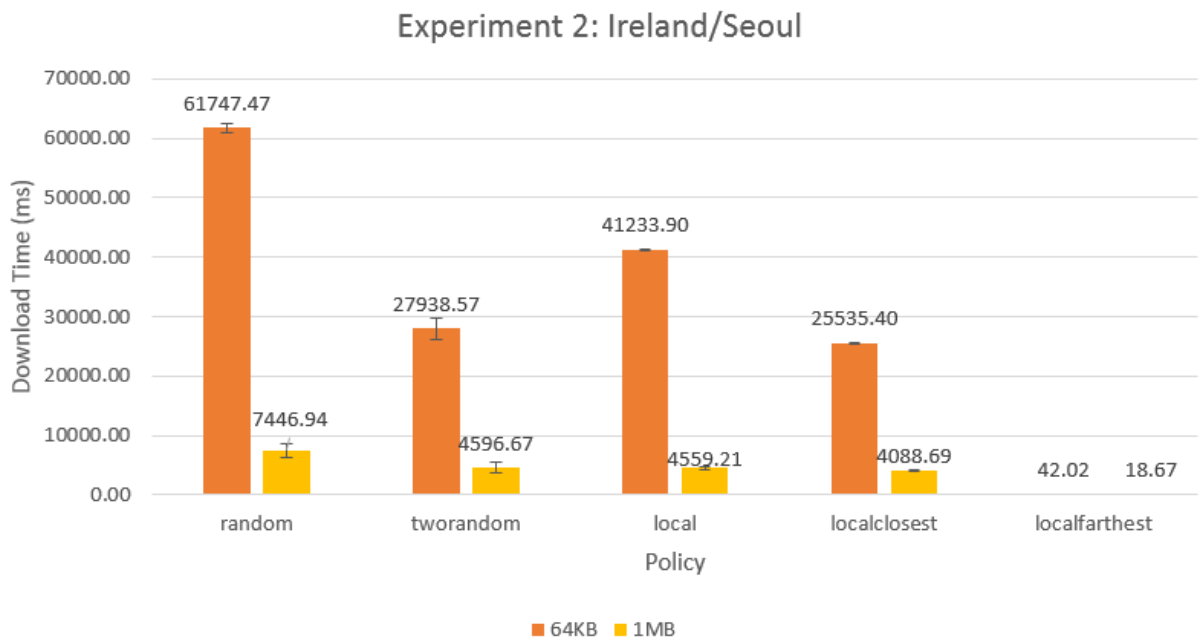


Figure 2. The above graph represents the results of three runs of Experiment 2, uploading from Ireland and downloading from Seoul. Each bar represents the average download time of 10 1MB files using various policies across 4 servers. The standard deviations are plotted as error bars.

For reference, we include the following 2 tables of the standard deviation values for Experiment 1 and Experiment 2 though they are already plotted as error bars on our graphs above. (Experiment 1 values are on the left, Experiment 2 on the right).

STD	64KB	1MB
random	717.175	329.467
tworandom	204.750	8.334
local	0.255	0.338
localclosest	0.512	0.251
localfarthest	0.735	0.041

STD	64KB	1MB
random	704.678	1037.764
tworandom	1886.816	899.924
local	138.492	234.884
localclosest	27.267	140.145
localfarthest	0.127	0.067

Figure 3. The standard deviations specified in table format for both experiments.

Note: The very first RTT of an experiment would occasionally be very high, resulting in our standard deviations for Dublin and Seoul sometimes being high despite very little frequency in any values other than that first one.

Findings and Conclusions

1. Correlation between observed RTT and throughput/bandwidth

There is a very strong assumed correlation between RTT and throughput/bandwidth. It is common sense that they should be inversely proportional. We can see this when we observe how quickly we are able to download data from a data center that is local and subsequently has a smaller RTT compared to when downloading from a datacenter further away (as shown in Figure 2 under policy local).

2. Relative performance of the policies for 1-site experiments.

The results from local, local closest, and local farthest policies all downloaded at about the same, shorter time. The results from random and two-random policies differed from both each other and from the other three policies. The random policy results in the largest download times, as expected, because the scattered blocks mean that a decent proportion of blocks will be located somewhere that takes longer to download from (in our case an example of such a location is Sao Paulo). The two-random policy decreases the time considerably because placing two copies of the block when only four servers exist greatly increases the chance that a block is local or very close. The local policy performs extremely well because a local download is extremely fast. The local closest policy performs the same as the local policy because the extra copy on the “closest server,” which in this case is Mumbai, is still farther than our local copy and so we download locally again. The same logic applies to the local farthest policy, where the extra copy is located in Sao Paulo (even farther from our local than Mumbai).

3. Relative performance of the policies for 2-site experiments.

The results vary greatly across the policies in this 2-site experiment. We can observe the same comparison between the random and two-random policies that we did in the 1-site experiment. This comparison is that the random policy takes the longest due to the placing of blocks in all four locations at random, and only having one copy (which forces us to download some proportion of blocks from the farthest center to us, Sao Paulo since we are downloading from Seoul). The second copy of the blocks in the two-random policy allows the RTT when fetching some blocks to be shorter due to more options when downloading blocks, resulting in a smaller download time. The local policy for our experiment uploads to Ireland, which we can see has one of the higher RTTs from Seoul, which causes our local policy experiment to have high download times. These download times are worse than the two random policy download times, which is expected since in the two random policy we can assume a large proportion (over the half we could assume in random) of the blocks are placed either locally or in Mumbai, which are both closer to Seoul than Ireland is. The local closest policy uploads to both Ireland and Mumbai, where Mumbai happens to be the closest datacenter to Seoul other than itself. This is why our local closest policy download times are faster than our local policy download times, and our two random policy times. Finally, our local farthest policy uploads to Ireland and to Seoul itself, so our downloads become local and run extremely fast.

4. Block size affecting the download time for each policy.

We can logically believe and see from our data that block size is inversely proportional to download time. As we are able to upload/download larger blocks, we have to make fewer round trips to gather all of our data and this is why our 1 MB experiments all have faster download times than their 64 KB counterparts.

5. Overall comments based on results

In this experiment, we learned that the policy we use to upload our data has drastically different results no matter which policy we use, but that the random policies have less variation across different data centers in comparison to local policies that become very “hit or miss”. We also learned that the RTTs between data centers is a very accurate depiction for how uploads/downloads between those data centers will occur.

6. Further experiments that could be performed.

An additional experiment we would like to run would be to run our uploader from Sao Paulo and download from Seoul to witness the local closest policy in comparison to the two random policy. We believe this would be interesting because local closest would upload to Sao Paulo and Ireland, which are both very far from Seoul. In this scenario, it is possible that local closest performs worse than two random.

Another additional experiment to perform could be to flood particular servers with requests and see at what point the client should choose to download the data from a server further away. From this we could draw comparisons between the locality of a server and its availability, finding at which point it makes sense to no longer rely on servers due to their proximity. It would be interesting to see what the effect of multiple downloads from a single server would have on its own ability to download and upload data.