How to Use the Performance Evaluation Program

The source code here evaluates the performance of a protocol that can verify the integrity of the data outsourced to a cloud. For the detailed protocol, please refer to our paper:

Fei Chen, Tao Xiang, Yuanyuan Yang, Cong Wang, Shengyu Zhang. Secure Cloud Storage Hits Distributed String Equality Checking: More Efficient, Conceptually Simpler, and Provably Secure. In Proc. of IEEE INFOCOM, 2015.

How to Run It

Basically, there are two ways to run the program. If you are using Eclipse for Java development, you can do it as follows:

1. Import our source code project into Eclipse. Try the menu “File -> Import -> Existing Projects into Workspace”.
2. Locate the file “PerformanceEvaluate.java” which is the entrance of the whole program.
3. Replace the string "D:\\test\\vs\\test5\\simplewiki-20130608-pages-meta-history.xml.7z" with your destination directory which stores the data that to be outsourced.
4. The performance result will be output in the console window.

If you are used to compile a java program in the command line, you can do it as follows:

1. Unzip the source code. Find all source codes in the directory “../src/fchen/”, compile all the source code.
2. Locate the file “PerformanceEvaluate.java” which is the entrance of the whole program.
3. Replace the string "D:\\test\\vs\\test5\\simplewiki-20130608-pages-meta-history.xml.7z" with your destination directory which stores the data that to be outsourced. Compile the program and then run it.
4. The performance result will be output in the console window.

Note that we employ a third-party utility class *MemoryUtil* to measure the size of a running-time object in the memory. To use this class, we need to send a parameter to the Java Virtual Machine: *-javaagent:classmexer.jar*. For more information about *MemoryUtil*, please refer to http://www.javamex.com/classmexer/.

Code Organization

There are 7 classes in the source code. For details, please refer to the help file, which is generated by the “javadoc” program. In the following, a brief introduction is presented.

1. InnerProductBasedVS – This is the main class that implements the proposed protocol. This class implements a protocol that employs the string equality check approach to solve the cloud storage integrity checking problem.
2. Benchmark.java – This class measures the performance of the proposed protocol in terms of memory and computation cost. It runs a lot of problem instances and then gets the performance data by averaging the experimental results.
3. PerformanceEvaluate.java – This is the main entrance for evaluating the performance of the protocol. To evaluate another data set to be outsourced, just modify this class, i.e. replacing the data set directory with another one.
4. CorrectnessTest.java – This class tests whether the prototype is correct.
5. KeyData – This class encapsulates all secret key material in one object.
6. ChallengeData.java –This class encapsulates the challenge data when auditing the cloud.
7. ProofData.java –This class encapsulates the proof data when the cloud proves that the storage is intact.