ECE297 Milestone 2

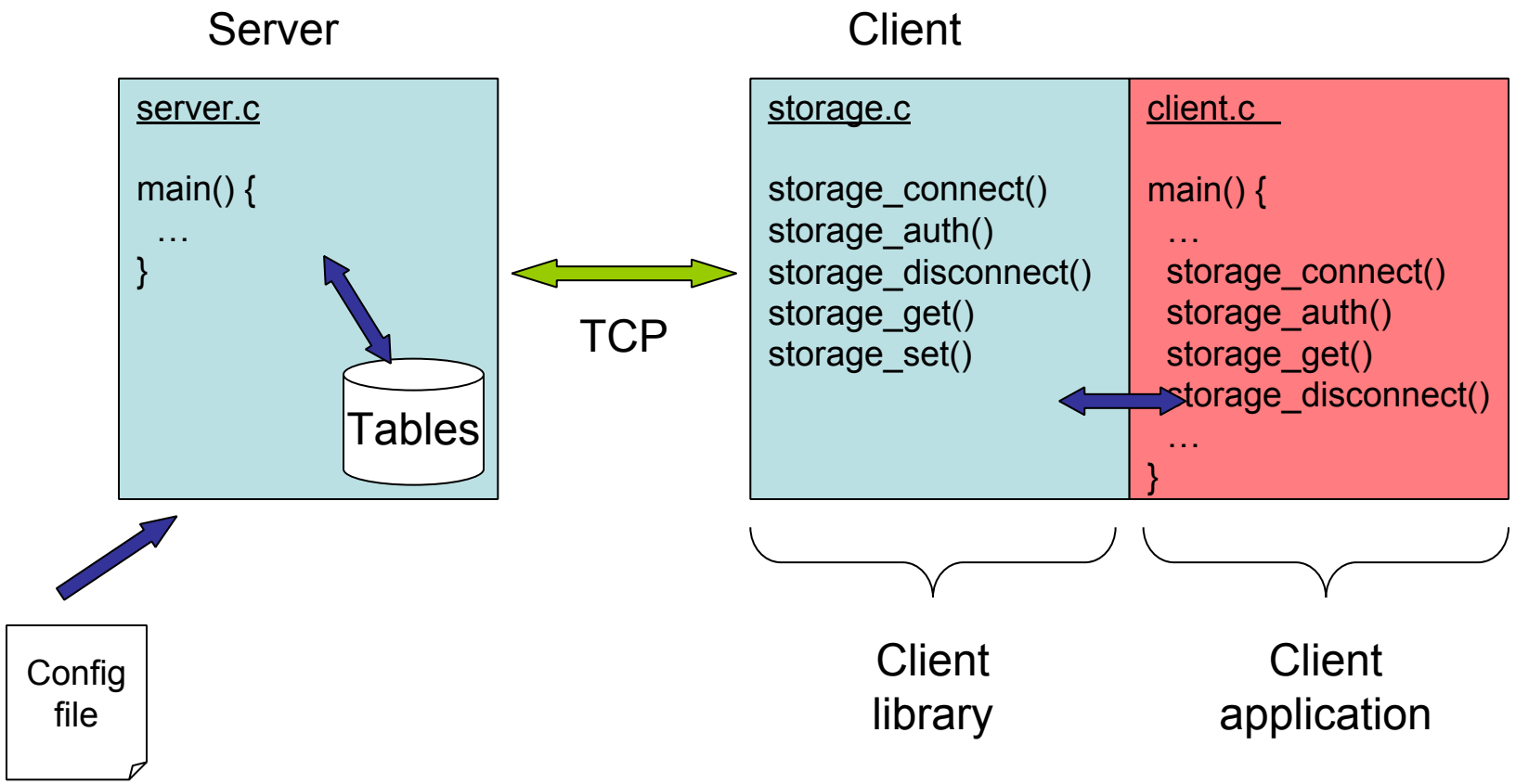
Jan. 29th, 2014

Payam Izadpanah (Original)
Djordje Maksimovic (Modified)

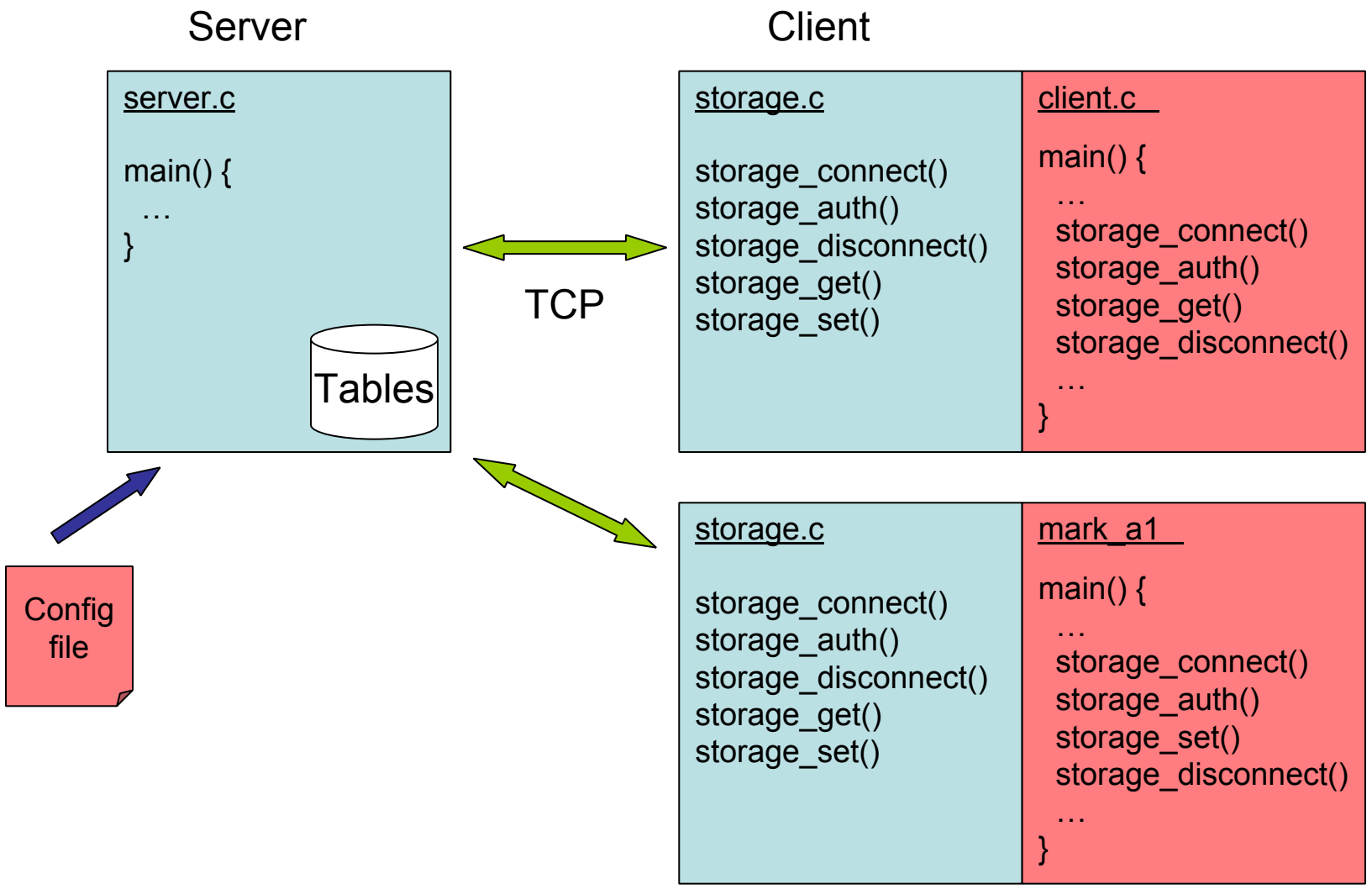
Agenda

- Architecture and implementation tasks
- Pre-submit test script
- Submitting the code

Architecture



Client apps use the client library



What to implement

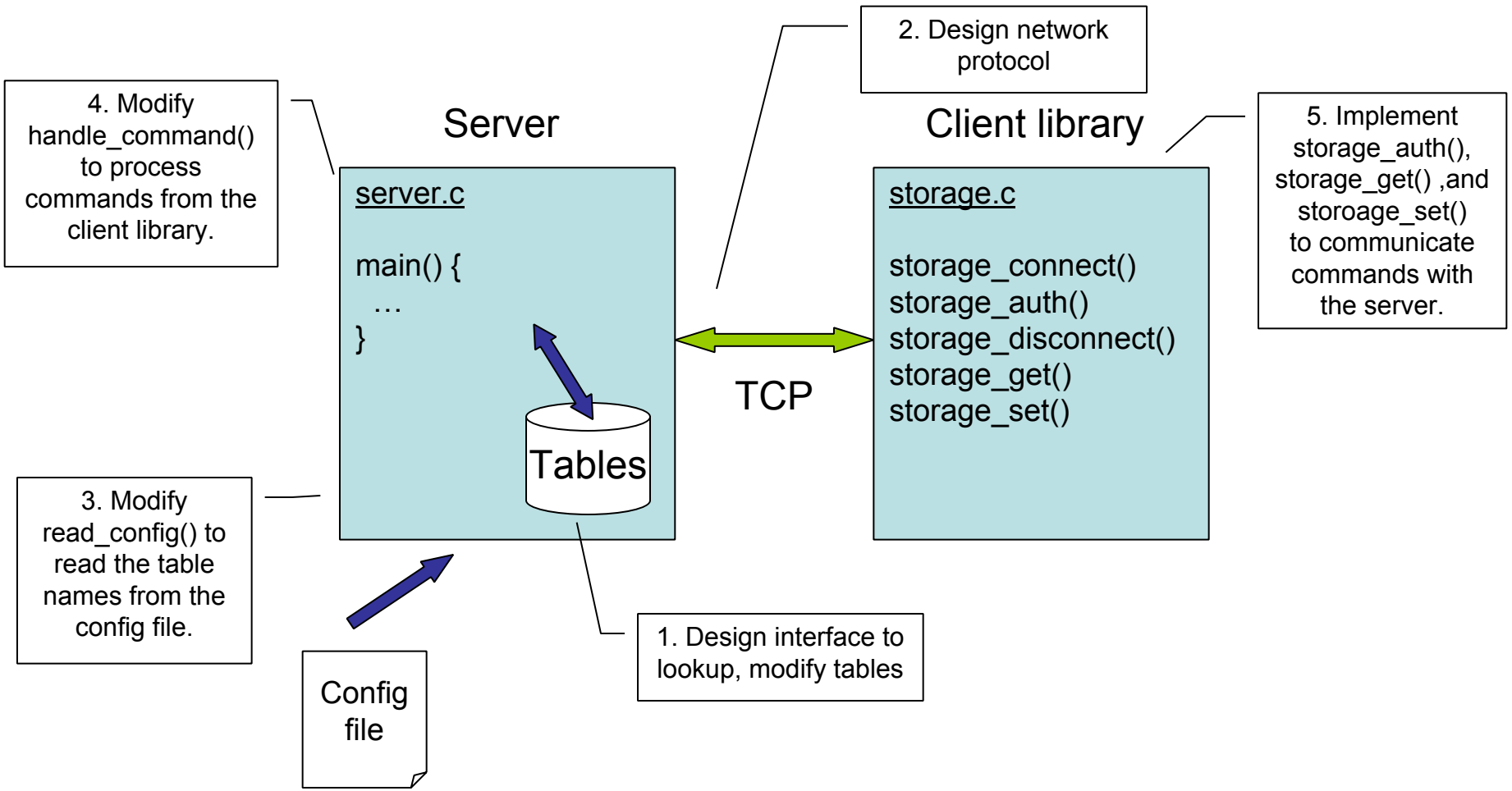


Table Format

- Table and key name are **alphanumeric**
- Key values are **alphanumeric and may contain spaces**
- **What kind of data structure will you use? (Pros/Cons)**
  - **Linked List**
  - **Binary Search Tree**
  - **Hash Table**
  - **Other/Mixture?**

Good Configuration File

```
server_host localhost
server_port 1111
username admin
password xxxnq.BMCifhU
table marks
```

Bad Configuration File

server_host localhost

server_port 1111

server_port 2222

username admin

table marks

Protocol Examples

- Text Based Protocols:
  - Set table=census key=toronto value="123 4"
    - Easy to read and write, can test protocol with netcat, but parser code will be larger, easy to extend
- Compact Protocols:
  - 3#census#toronto#123 45\n
    - Easy to parse, may not be easy to read, may be difficult to extend

Performance evaluation

- Measure the performance of the server under three configurations:
  - A configuration without logging,
  - logging enabled to *stdout*, and
  - logging enabled to a *file*.
- Metrics:
  - End-to-end execution time
  - Server processing time

Census Table

- The table name should be census.
- The key is the name of the city region.
- The value is the Population count of the city as of 2006. (An integer)
- The table should be populated with the data from the census subdivisions.
- You should scrub the raw data to get the city and population

  3520005,"Toronto (Ont.)",C ,F,2503281,2481494,F,F,0.9,1040597,979330,630.1763,3972.4,1

- you should strip any special characters in the city names, such as spaces or dashes

Some points to remember

- Begin from one of your team members' milestone 1 code
- Don't worry about multiple clients
- Don't change storage.h
- The code must be written in C
- The server must be buildable by running make server in the src directory.
- The server executable must be called server

Some points to remember

- The client library must be buildable by running make libstorage.a in the src directory.
- The client library must be called libstorage.a
- Some useful size limits are in storage.h
  - E.g., MAX_TABLES, MAX_KEY_LEN

Pre-submit test

Pre-submit test

- Run the testsubmit script on your submission before you submit
- Script will
  - Check for required submission files
  - Verify directory structure within the .tgz files
  - Build your code
  - Run a few test cases on your code
  - …
- Some errors are fatal (no .tgz or .diff file), others are warnings (e.g., test cases fail)
- **Script does not submit your code**

Running the testsubmit script

- Create the three submission files in ~/ece297/submit
  - storage-asst2.tgz
  - storage-asst2.diff
  - doxygen-asst2.tgz

- Run testsubmit script: >/cad2/ece297s/public/testsubmitece297s 2
- Wait (several minutes) for test cases to finish
- Save the output of testsubmit script >/cad2/ece297s/public/testsubmitece297s 2 > testsubmitAss2.log
- Include the output file testsubmitAss2.log in storage-asst2.tgz (required to be submitted)

Notes on the testsubmit script

- Don’t wait until the last minute to run the testsubmit script
- Get in the habit of using SVN now
  - The testsubmit script will fail if you simply try to submit your working directory
- There is no harm in running it as often as you want
- Practice submission steps now (even with the skeleton code)!
- Our auto-marking tests are similar to the testsubmit script
  - If your code fails the testsubmit tests, it will likely fail the marking tests as well
- The testsubmit script only partially verifies your submission
  - You should still follow the instructions in the handout

Submitting the code

Submitting the Code

- The three required files
  - storage-asst2.tgz
  - storage-asst2.diff
  - doxygen-asst2.tgz

- Performance report
  - performance-asst2.pdf

  (see the instruction on the course website for creating and submitting these files)

- Your census data should be available at storage/data/census/

- storage/data/census.conf should be included in the submitted storage-asst2.tgz

Submitting the Code

- See the submitted file (check the size and date)

  > submitece297s -l 2

- **We only mark the most recent files submitted by any member of the teams.**
- Check these files by entering:

  > checksubmit 2

- Check the files' size, date, and the user who submitted the files.
- Ensure logging=0 for both server and client

Submitting the Code

- Some tests are available

  /cad2/ece297s/public/assignment2/a2-partial.tgz.

  You can try running them as follows:

  > cd ~/ece297/storage/test

  > tar zxf /cad2/ece297s/public/assignment2/a2-partial.tgz

  > cd a2-partial

  > make clean run

Submitting the code

- Final Remark
  - **Make sure your submitted code passes the test suites given to you as part of the skeleton distribution and that it passed the testsubmitece297s test before submission.**

Partial marks

- Some implementation attempt and code that builds
- Client library only
  - Check for some error conditions (e.g., invalid parameters)
- Client library and server communication
- Client library and server communication with table management at the server