

# COMP 8005

## Project

## Testing

Harman Dhillon  
A00994245  
Dec 3, 2025

<b>Tests</b>	<b>6</b>
Test 1	6
Test 2	7
Test 3	8
Test 4	9
Test 5	10
Test 6	11
Test 7	12
Test 8	13
Test 9	14
Test 10	15
Test 11	16
Test 12	17
Test 13	18
Test 14	19
Test 15	20
Test 16	20
Test 17	21
Test 18	21

Test	Expected	Actual	Screenshot
Run the program with no arguments  <b>Command:</b> <code>./client</code>	fail	fail	<a href="#">Test 1</a>
Run the program with only -s flag  <b>Command:</b> <code>./client -s</code>	fail	fail	<a href="#">Test 2</a>
Run the program with only -s flag and the ip address  <b>Command:</b> <code>./client -s 192.168.1.67</code>	fail	fail	<a href="#">Test 3</a>
Run the program with the -s flag and the ip address and the -p flag  Command: <code>./client -s 192.168.1.67 -p</code>	fail	fail	<a href="#">Test 4</a>
Run the program with the -s flag and the ip address and the -p flag with the port number when server is running  Command: <code>./client -s 192.168.1.67 -p 34000</code>	pass	pass	<a href="#">Test 5</a>
Run the program with the -s flag and the ip address and the -p flag with the port number and the -t flag  Command: <code>./client -s 192.168.1.67 -p 34000 -t</code>	fail	fail	<a href="#">Test 6</a>
Run the program with the -s flag and the ip address and the -p flag with the port number and the -t flag with number of threads when server is running  Command: <code>./client -s 192.168.1.67 -p 34000 -t 4</code>	pass	pass	<a href="#">Test 7</a>
Run the program with the -s flag and the ip address and the -p flag with the port number and the -t flag with number of threads while the server is not running  Command: <code>./client -s 192.168.1.67 -p 34000 -t 4</code>	fail	fail	<a href="#">Test 8</a>
Run the program with no arguments	fail	fail	<a href="#">Test 9</a>

Command: ./server			
Run the program with only -s flag  Command: ./server -s	fail	fail	<a href="#">Test 10</a>
Run the program with only -s flag and the ip address  Command: ./server -s 192.168.1.67	fail	fail	<a href="#">Test 11</a>
Run the program with the -s flag and the ip address and the -p flag  Command: ./server -s 192.168.1.67 -p	fail	fail	<a href="#">Test 12</a>
Run the program with the -s flag and the ip address and the -p flag with the port number  Command: ./server -s 192.168.1.67 -p 34000	fail	fail	<a href="#">Test 13</a>
Run the program with the -s flag and the ip address and the -p flag with the port number and the -H flag  Command: ./server -s 192.168.1.67 -p 34000 -H	fail	fail	<a href="#">Test 14</a>
Run the program with the -s flag and the ip address and the -p flag with the port number and the -H flag with the hash  Command: ./server -s 192.168.1.67 -p 34000 -H '\$2b\$12\$cNPD3sslWWNg2BkhS4eSD.KTC.6Y878H6MM HkbceIGNKcYV35nC8m'	pass	pass	<a href="#">Test 15</a>
Server reclaims work when client disconnects  <b>Command:</b>	pass	pass	<a href="#">Test 16</a>
When worker times out, server reclaims work	pass	pass	<a href="#">Test 17</a>
Assign reclaimed work to the next worker available	pass	pass	<a href="#">Test 18</a>

# Tests

## Test 1

```
.../client/build main • ➤ ./client
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 104.

TRACE: STATE_HANDLE_ARGUMENTS
Entered state at line 116.

Usage: ./client [OPTIONS]

Required options:
  -s, --server <addr>      Server IP address or hostname (required)
  -p, --port <num>          Server listen port (required)
Optional options:
  -t, --threads <num>       Number of threads the worker will use
                            (default: 4)
  -h, --help                Display this help message and exit

Examples:
./client --server 192.168.1.10 --port 5000
./client -s example.com -p 5000 -t 8

Notes:
• Long and short forms may be used interchangeably (e.g. --port or -p).
• If threads is omitted it defaults to 4.
• The program will validate numeric ranges (e.g. port must fit in uint16).
ERROR The server IP address is required.
In file command_line.c in function handle_arguments on line 134
TRACE: STATE_CLEANUP
Entered state at line 258.
```

## Test 2

```
.../client/build main • ➤ ./client -s
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 104.

Usage: ./client [OPTIONS]

Required options:
  -s, --server <addr>      Server IP address or hostname (required)
  -p, --port <num>          Server listen port (required)
Optional options:
  -t, --threads <num>       Number of threads the worker will use
                            (default: 4)
  -h, --help                 Display this help message and exit

Examples:
  ./client --server 192.168.1.10 --port 5000
  ./client -s example.com -p 5000 -t 8

Notes:
  • Long and short forms may be used interchangeably (e.g. --port or -p).
  • If threads is omitted it defaults to 4.
  • The program will validate numeric ranges (e.g. port must fit in uint16).
ERROR Unknown option '-s'.
In file command_line.c in function parse_arguments on line 85
TRACE: STATE_CLEANUP
Entered state at line 258.
```

## Test 3

```
.../client/build main • ➤ ./client -s 192.168.1.67
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 104.

TRACE: STATE_HANDLE_ARGUMENTS
Entered state at line 116.

Usage: ./client [OPTIONS]

Required options:
  -s, --server <addr>      Server IP address or hostname (required)
  -p, --port <num>          Server listen port (required)
Optional options:
  -t, --threads <num>       Number of threads the worker will use
                             (default: 4)
  -h, --help                 Display this help message and exit

Examples:
  ./client --server 192.168.1.10 --port 5000
  ./client -s example.com -p 5000 -t 8

Notes:
  • Long and short forms may be used interchangeably (e.g. --port or -p).
  • If threads is omitted it defaults to 4.
  • The program will validate numeric ranges (e.g. port must fit in uint16).
ERROR The server port is required.
In file command_line.c in function handle_arguments on line 142
TRACE: STATE_CLEANUP
Entered state at line 258.
```

## Test 4

```
.../client/build main • ➤ ./client -s 192.168.1.67 -p
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 104.

Usage: ./client [OPTIONS]

Required options:
  -s, --server <addr>      Server IP address or hostname (required)
  -p, --port <num>          Server listen port (required)
Optional options:
  -t, --threads <num>       Number of threads the worker will use
                             (default: 4)
  -h, --help                 Display this help message and exit

Examples:
  ./client --server 192.168.1.10 --port 5000
  ./client -s example.com -p 5000 -t 8

Notes:
  • Long and short forms may be used interchangeably (e.g. --port or -p).
  • If threads is omitted it defaults to 4.
  • The program will validate numeric ranges (e.g. port must fit in uint16).
ERROR Unknown option '-p'.
In file command_line.c in function parse_arguments on line 85
TRACE: STATE_CLEANUP
Entered state at line 258.
```

## Test 5

```
.../client/build main • ➤ ./client -s 192.168.1.67 -p 34000
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 104.

TRACE: STATE_HANDLE_ARGUMENTS
Entered state at line 116.

TRACE: STATE_CONVERT_ADDRESS
Entered state at line 129.

TRACE: STATE_CREATE_SOCKET
Entered state at line 142.

TRACE: STATE_CONNECT_SOCKET
Entered state at line 156.

Connecting to: 192.168.1.67:34000
Connected to: 192.168.1.67:34000
TRACE: STATE_WAIT_HASH
Entered state at line 169.

[WORKER] Received hash: $2b$12$cNPD3sslWwNg2BkhS4eSD.KTC.6Y878H6MMHkbceIGNKcYV35nC8m
TRACE: STATE_WAIT_WORK
Entered state at line 182.

Buffer: WORK 0 7000 100 600

[WORKER] Received WORK: start=0, len=7000, checkpoint=100, timeout=600, end index: 6999
TRACE: STATE_START_TIMER
Entered state at line 201.

TRACE: STATE_START_CRACKING
Entered state at line 214.
```

## Test 6

```
.../client/build main • ✘ ./client -s 192.168.1.67 -p 34000 -t
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 104.

Usage: ./client [OPTIONS]

Required options:
  -s, --server <addr>      Server IP address or hostname (required)
  -p, --port <num>          Server listen port (required)
Optional options:
  -t, --threads <num>       Number of threads the worker will use
                             (default: 4)
  -h, --help                 Display this help message and exit

Examples:
  ./client --server 192.168.1.10 --port 5000
  ./client -s example.com -p 5000 -t 8

Notes:
  • Long and short forms may be used interchangeably (e.g. --port or -p).
  • If threads is omitted it defaults to 4.
  • The program will validate numeric ranges (e.g. port must fit in uint16).
ERROR Unknown option '-t'.
In file command_line.c in function parse_arguments on line 85
TRACE: STATE_CLEANUP
Entered state at line 258.
```

## Test 7

```
.../client/build main • ➤ ./client -s 192.168.1.67 -p 34000 -t 4
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 104.

TRACE: STATE_HANDLE_ARGUMENTS
Entered state at line 116.

TRACE: STATE_CONVERT_ADDRESS
Entered state at line 129.

TRACE: STATE_CREATE_SOCKET
Entered state at line 142.

TRACE: STATE_CONNECT_SOCKET
Entered state at line 156.

Connecting to: 192.168.1.67:34000
Connected to: 192.168.1.67:34000
TRACE: STATE_WAIT_HASH
Entered state at line 169.

[WORKER] Received hash: $2b$12$cNPD3sslWWNg2BkhS4eSD.KTC.6Y878H6MMHkbceIGNKcYV35nC8m
TRACE: STATE_WAIT_WORK
Entered state at line 182.

Buffer: WORK 0 7000 100 600

[WORKER] Received WORK: start=0, len=7000, checkpoint=100, timeout=600, end index: 6999
TRACE: STATE_START_TIMER
Entered state at line 201.

TRACE: STATE_START_CRACKING
Entered state at line 214.
```

## Test 8

```
.../client/build main • ➤ ./client -s 192.168.1.67 -p 34000 -t 4
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 104.

TRACE: STATE_HANDLE_ARGUMENTS
Entered state at line 116.

TRACE: STATE_CONVERT_ADDRESS
Entered state at line 129.

TRACE: STATE_CREATE_SOCKET
Entered state at line 142.

TRACE: STATE_CONNECT_SOCKET
Entered state at line 156.

Connecting to: 192.168.1.67:34000
ERROR Connection refused
In file server_config.c in function socket_connect on line 41
TRACE: STATE_CLEANUP
Entered state at line 258.
```

## Test 9

```
.../server/build main • ➤ ./server
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 92.

TRACE: STATE_HANDLE_ARGUMENTS
Entered state at line 104.

Usage: ./server [OPTIONS]

Required options:
  -s, --server <addr>      Server IP address or hostname (required)
  -p, --port <num>          Server listen port (required)
  -H, --hash <hash>         Hashed password to crack (required)

Optional options:
  -w, --work-size <num>    Number of passwords assigned per node request
                            (default: 1000)
  -c, --checkpoint <num>   Number of attempts before a node sends a checkpoint
                            (default: work-size / 4)
  -t, --timeout <num>     Seconds to wait for a checkpoint from a client
                            (default: 600)
  -h, --help                Display this help message and exit

Examples:
  ./server --server 192.168.1.10 --port 5000 --hash $6$... --work-size 1000
  ./server -s example.com -p 5000 -H <hash> -c 500 -t 300

Notes:
  • Long and short forms may be used interchangeably (e.g. --port or -p).
  • If work-size is omitted it defaults to 1000.
  • If checkpoint is omitted it defaults to work-size / 4.
  • The program will validate numeric ranges (e.g. port must fit in uint16).

ERROR The server IP address is required.
In file command_line.c in function handle_arguments on line 193
TRACE: STATE_CLEANUP
Entered state at line 234.
```

## Test 10

```
.../server/build main • ➤ ./server -s
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 92.

Usage: ./server [OPTIONS]

Required options:
  -s, --server <addr>      Server IP address or hostname (required)
  -p, --port <num>          Server listen port (required)
  -H, --hash <hash>         Hashed password to crack (required)

Optional options:
  -w, --work-size <num>    Number of passwords assigned per node request
                            (default: 1000)
  -c, --checkpoint <num>   Number of attempts before a node sends a checkpoint
                            (default: work-size / 4)
  -t, --timeout <num>     Seconds to wait for a checkpoint from a client
                            (default: 600)
  -h, --help                Display this help message and exit

Examples:
  ./server --server 192.168.1.10 --port 5000 --hash $6$... --work-size 1000
  ./server -s example.com -p 5000 -H <hash> -c 500 -t 300

Notes:
  • Long and short forms may be used interchangeably (e.g. --port or -p).
  • If work-size is omitted it defaults to 1000.
  • If checkpoint is omitted it defaults to work-size / 4.
  • The program will validate numeric ranges (e.g. port must fit in uint16).
ERROR Unknown option '-s'.
In file command_line.c in function parse_arguments on line 138
TRACE: STATE_CLEANUP
Entered state at line 234.
```

## Test 11

```
.../server/build main • ➤ ./server -s 192.168.1.67
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 92.

TRACE: STATE_HANDLE_ARGUMENTS
Entered state at line 104.

Usage: ./server [OPTIONS]

Required options:
-s, --server <addr>      Server IP address or hostname (required)
-p, --port <num>           Server listen port (required)
-H, --hash <hash>          Hashed password to crack (required)

Optional options:
-w, --work-size <num>     Number of passwords assigned per node request
                           (default: 1000)
-c, --checkpoint <num>    Number of attempts before a node sends a checkpoint
                           (default: work-size / 4)
-t, --timeout <num>       Seconds to wait for a checkpoint from a client
                           (default: 600) ┌
-h, --help                  Display this help message and exit

Examples:
./server --server 192.168.1.10 --port 5000 --hash $6$... --work-size 1000
./server -s example.com -p 5000 -H <hash> -c 500 -t 300

Notes:
• Long and short forms may be used interchangeably (e.g. --port or -p).
• If work-size is omitted it defaults to 1000.
• If checkpoint is omitted it defaults to work-size / 4.
• The program will validate numeric ranges (e.g. port must fit in uint16).

ERROR The server port is required.
In file command_line.c in function handle_arguments on line 201
TRACE: STATE_CLEANUP
Entered state at line 234.
```

## Test 12

```
.../server/build main • ➤ ./server -s 192.168.1.67 -p
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 92.

Usage: ./server [OPTIONS]

Required options:
  -s, --server <addr>      Server IP address or hostname (required)
  -p, --port <num>          Server listen port (required)
  -H, --hash <hash>         Hashed password to crack (required)

Optional options:
  -w, --work-size <num>    Number of passwords assigned per node request
                            (default: 1000)
  -c, --checkpoint <num>   Number of attempts before a node sends a checkpoint
                            (default: work-size / 4)
  -t, --timeout <num>     Seconds to wait for a checkpoint from a client
                            (default: 600)
  -h, --help                Display this help message and exit

Examples:
  ./server --server 192.168.1.10 --port 5000 --hash $6$... --work-size 1000
  ./server -s example.com -p 5000 -H <hash> -c 500 -t 300

Notes:
  • Long and short forms may be used interchangeably (e.g. --port or -p).
  • If work-size is omitted it defaults to 1000.
  • If checkpoint is omitted it defaults to work-size / 4.
  • The program will validate numeric ranges (e.g. port must fit in uint16).
ERROR Unknown option '-p'.
In file command_line.c in function parse_arguments on line 138
TRACE: STATE_CLEANUP
Entered state at line 234.
```

## Test 13

```
.../server/build main • ➤ ./server -s 192.168.1.67 -p 34000
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 92.

TRACE: STATE_HANDLE_ARGUMENTS
Entered state at line 104.

Usage: ./server [OPTIONS]

Required options:
  -s, --server <addr>      Server IP address or hostname (required)
  -p, --port <num>          Server listen port (required)
  -H, --hash <hash>         Hashed password to crack (required)

Optional options:
  -w, --work-size <num>    Number of passwords assigned per node request
                            (default: 1000)
  -c, --checkpoint <num>   Number of attempts before a node sends a checkpoint
                            (default: work-size / 4)
  -t, --timeout <num>     Seconds to wait for a checkpoint from a client
                            (default: 600)
  -h, --help                Display this help message and exit

Examples:
  ./server --server 192.168.1.10 --port 5000 --hash $6$... --work-size 1000
  ./server -s example.com -p 5000 -H <hash> -c 500 -t 300

Notes:
  • Long and short forms may be used interchangeably (e.g. --port or -p).
  • If work-size is omitted it defaults to 1000.
  • If checkpoint is omitted it defaults to work-size / 4.
  • The program will validate numeric ranges (e.g. port must fit in uint16).
ERROR The Hash is required!
In file command_line.c in function handle_arguments on line 209
TRACE: STATE_CLEANUP
Entered state at line 234.
```

## Test 14

```
.../server/build main • ➤ ./server -s 192.168.1.67 -p 34000 -H
TRACE: STATE_PARSE_ARGUMENTS
Entered state at line 92.

Usage: ./server [OPTIONS]

Required options:
  -s, --server <addr>      Server IP address or hostname (required)
  -p, --port <num>          Server listen port (required)
  -H, --hash <hash>         Hashed password to crack (required)

Optional options:
  -w, --work-size <num>    Number of passwords assigned per node request
                           (default: 1000)
  -c, --checkpoint <num>   Number of attempts before a node sends a checkpoint
                           (default: work-size / 4)
  -t, --timeout <num>     Seconds to wait for a checkpoint from a client
                           (default: 600)
  -h, --help                Display this help message and exit

Examples:
  ./server --server 192.168.1.10 --port 5000 --hash $6$... --work-size 1000
  ./server -s example.com -p 5000 -H <hash> -c 500 -t 300

Notes:
  • Long and short forms may be used interchangeably (e.g. --port or -p).
  • If work-size is omitted it defaults to 1000.
  • If checkpoint is omitted it defaults to work-size / 4.
  • The program will validate numeric ranges (e.g. port must fit in uint16).
ERROR Unknown option '-H'.
In file command_line.c in function parse_arguments on line 138
TRACE: STATE_CLEANUP
Entered state at line 234.
```

## Test 15

```
.../server/build main • > ./server -s 192.168.1.67 -p 34000 -H '$2b$12$cNPD3sslWWNg2BkhS4eSD.KTC.6Y878H6MMHkbceIGNKcYV35nC8m'  
TRACE: STATE_PARSE_ARGUMENTS  
Entered state at line 92.  
  
TRACE: STATE_HANDLE_ARGUMENTS  
Entered state at line 104.  
  
TRACE: STATE_CONVERT_ADDRESS  
Entered state at line 117.  
  
TRACE: STATE_CREATE_SOCKET  
Entered state at line 129.  
  
TRACE: STATE_BIND_SOCKET  
Entered state at line 143.  
  
binding to: 192.168.1.67:34000  
Bound to socket: 192.168.1.67:34000  
TRACE: STATE_START_LISTENING  
Entered state at line 156.  
  
TRACE: STATE_START_TIMER  
Entered state at line 188.  
  
TRACE: STATE_START_POLLING  
Entered state at line 198.
```

## Test 16

## Test 17

```
Entered state at line 143.  
binding to: 192.168.1.67:34000  
Bound to socket: 192.168.1.67:34000  
TRACE: STATE_START_LISTENING  
Entered state at line 156.  
TRACE: STATE_START_TIMER  
Entered state at line 188.  
TRACE: STATE_START_POLLING  
Entered state at line 190.  
^CTRACE: STATE_STOP_TIMER  
Entered state at line 217.  
Total time workers spent: 0 seconds  
Server ran for: 1.18 seconds  
TRACE: STATE_CLEANUP  
Entered state at line 234.  
  
./server/build main • ? > ./server -s 192.168.1.67 -p 34000 -H '$2b$12$2dR27n7o/08XoEdRvbSRuczFHPQiuRutEsMloV3.  
6rjptKmQaSPu!' -w 1000 -c 250 -t 1  
TRACE: STATE_PARSE_ARGUMENTS  
Entered state at line 92.  
TRACE: STATE_HANDLE_ARGUMENTS  
Entered state at line 104.  
TRACE: STATE_CONVERT_ADDRESS  
Entered state at line 117.  
TRACE: STATE_CREATE_SOCKET  
Entered state at line 129.  
TRACE: STATE_BIND_SOCKET  
Entered state at line 143.  
binding to: 192.168.1.67:34000  
Bound to socket: 192.168.1.67:34000  
TRACE: STATE_START_LISTENING  
Entered state at line 156.  
TRACE: STATE_START_TIMER  
Entered state at line 188.  
TRACE: STATE_START_POLLING  
Entered state at line 198.  
Connected to client: 5  
[SERVER] Sent HASH to worker(fd=5)  
[SERVER] Worker 5 is READY  
[SERVER] Assigned worker(fd=5) work: start=0, size=1000, checkpoint=250, timeout=1  
[SERVER] Worker 5 checkpoint = 0  
Worker taking over assigned work  
[SERVER] Reclaiming 1000 units of unfinished work from 5 (0 -> 999)  
■  
  
Distributed Password Cracker/data main • ? > cd ../source/client/build/  
./client/build main • ? > ./client -s 192.168.1.67 -p 34000 -t 4  
TRACE: STATE_PARSE_ARGUMENTS  
Entered state at line 104.  
TRACE: STATE_HANDLE_ARGUMENTS  
Entered state at line 116.  
TRACE: STATE_CONVERT_ADDRESS  
Entered state at line 129.  
TRACE: STATE_CREATE_SOCKET  
Entered state at line 142.  
TRACE: STATE_CONNECT_SOCKET  
Entered state at line 156.  
Connecting to: 192.168.1.67:34000  
Connected to: 192.168.1.67:34000  
TRACE: STATE_WAIT_HASH  
Entered state at line 169.  
[WORKER] Received hash: $2b$12$2dR27n7o/08XoEdRvbSRuczFHPQiuRutEsMloV3.6rjptKmQaSPu!  
TRACE: STATE_WAIT_WORK  
Entered state at line 182.  
Buffer: WORK 0 1000 250 1  
[WORKER] Received WORK: start=0, len=1000, checkpoint=250, timeout=1, end index: 999  
TRACE: STATE_START_TIMER  
Entered state at line 201.  
TRACE: STATE_START_CRACKING  
Entered state at line 214.  
Sent checkpoint 0  
Sent checkpoint 250  
TRACE: STATE_STOP_TIMER  
Entered state at line 240.  
Wall: 17.320614 s  
CPU: 69.032888 s  
TRACE: STATE_CLEANUP  
Entered state at line 258.  
■  
./client/build main • ? > []
```

## Test 18

```
[SERVER] Reclaiming 3250 units of unfinished work from 5 (3250 -> 6499)  
[SERVER] Worker 6 checkpoint → 9750  
[SERVER] Worker 6 finished its work in 6 seconds.  
[SERVER] Assigned worker(fd=6) work: start=3250, size=3250, checkpoint=3250, timeout=600
```