

Midterm Project – Chord

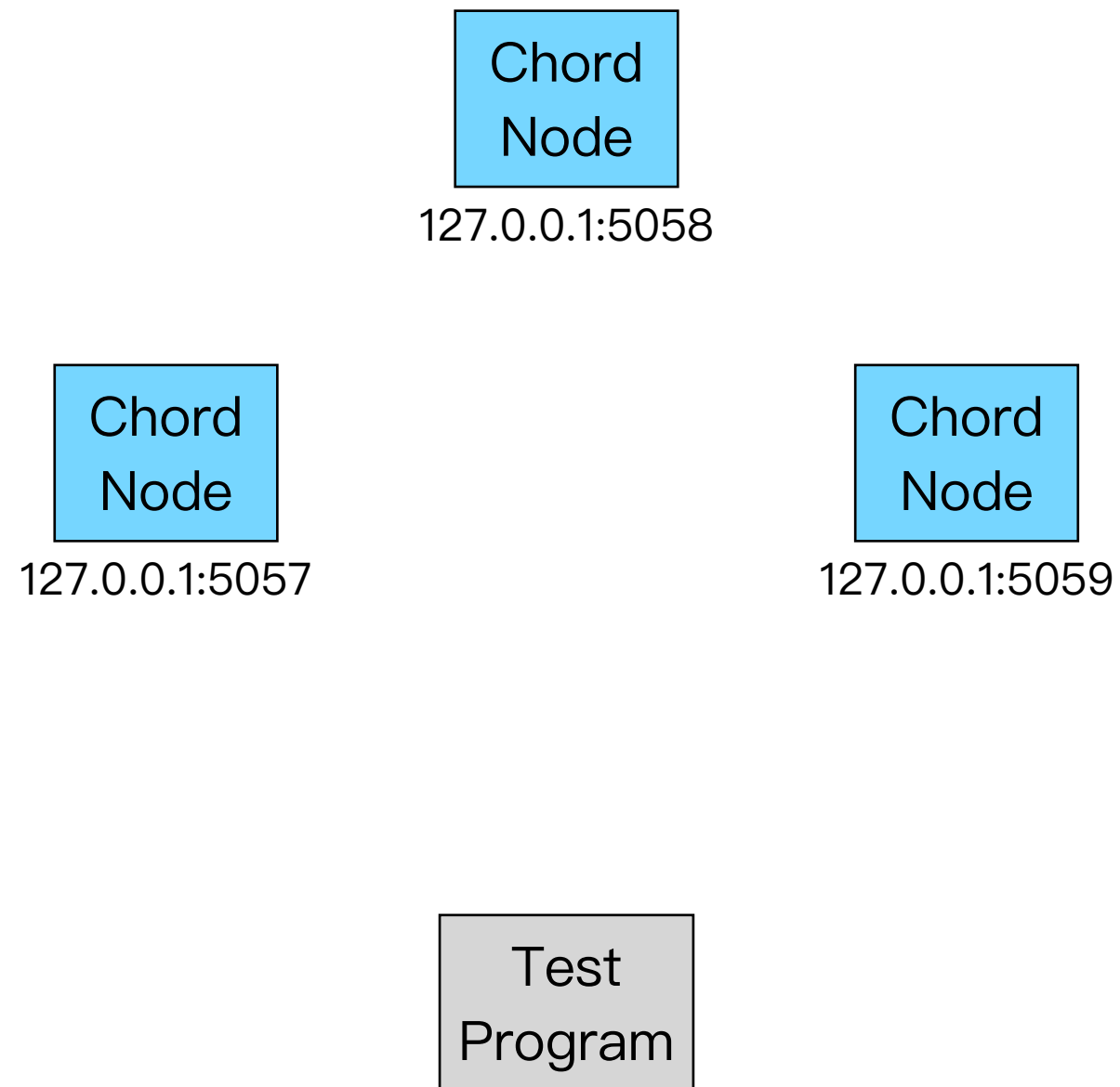
Part 1: DHT Layer Implementation

Practices for Distributed Systems and Cloud Application
Development

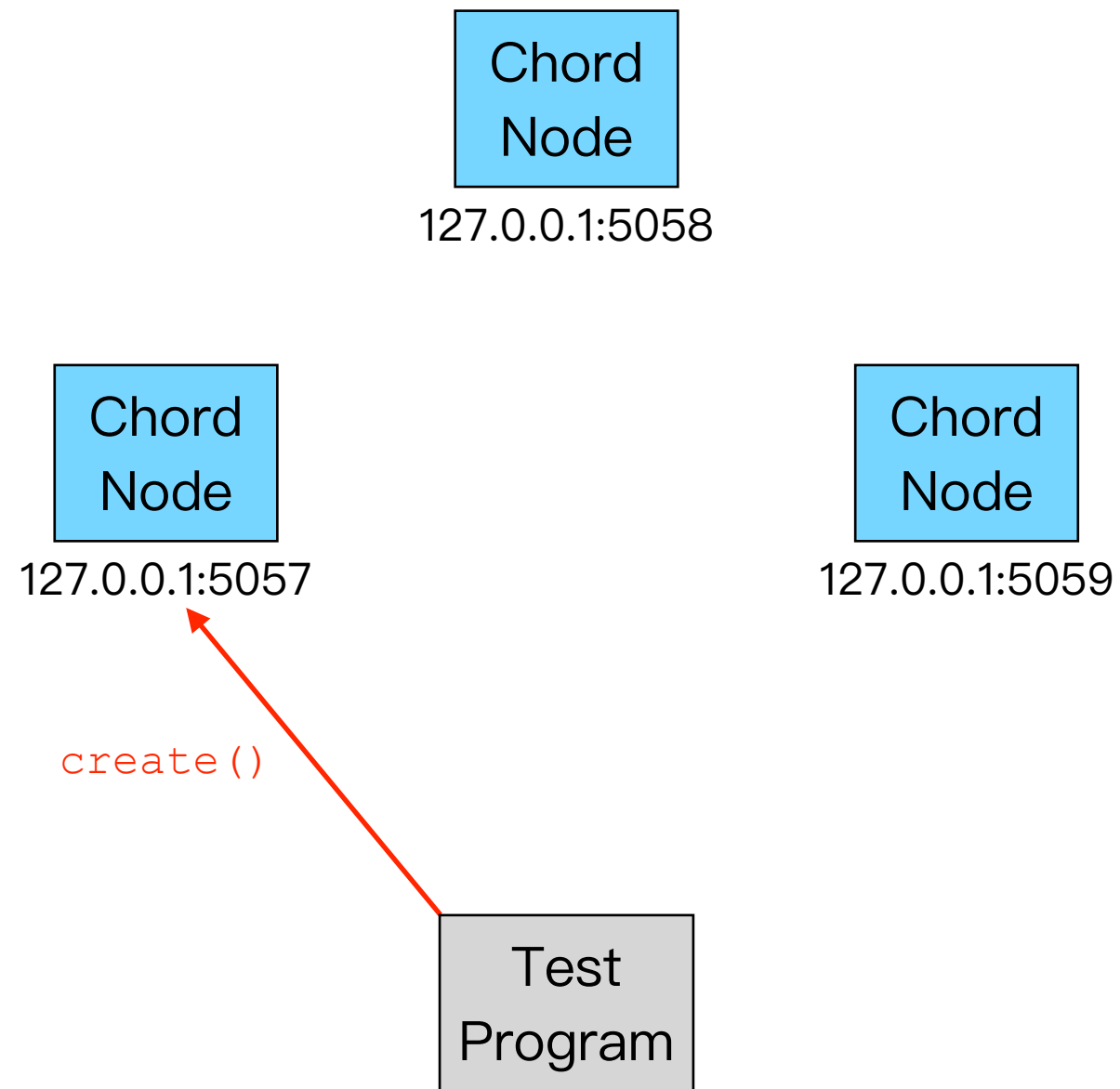
Goal

- Implement Chord node in C++ using rpclib
- Invoke the RPC methods of Chord nodes using Python MessagePack RPC
- Test your implementation locally

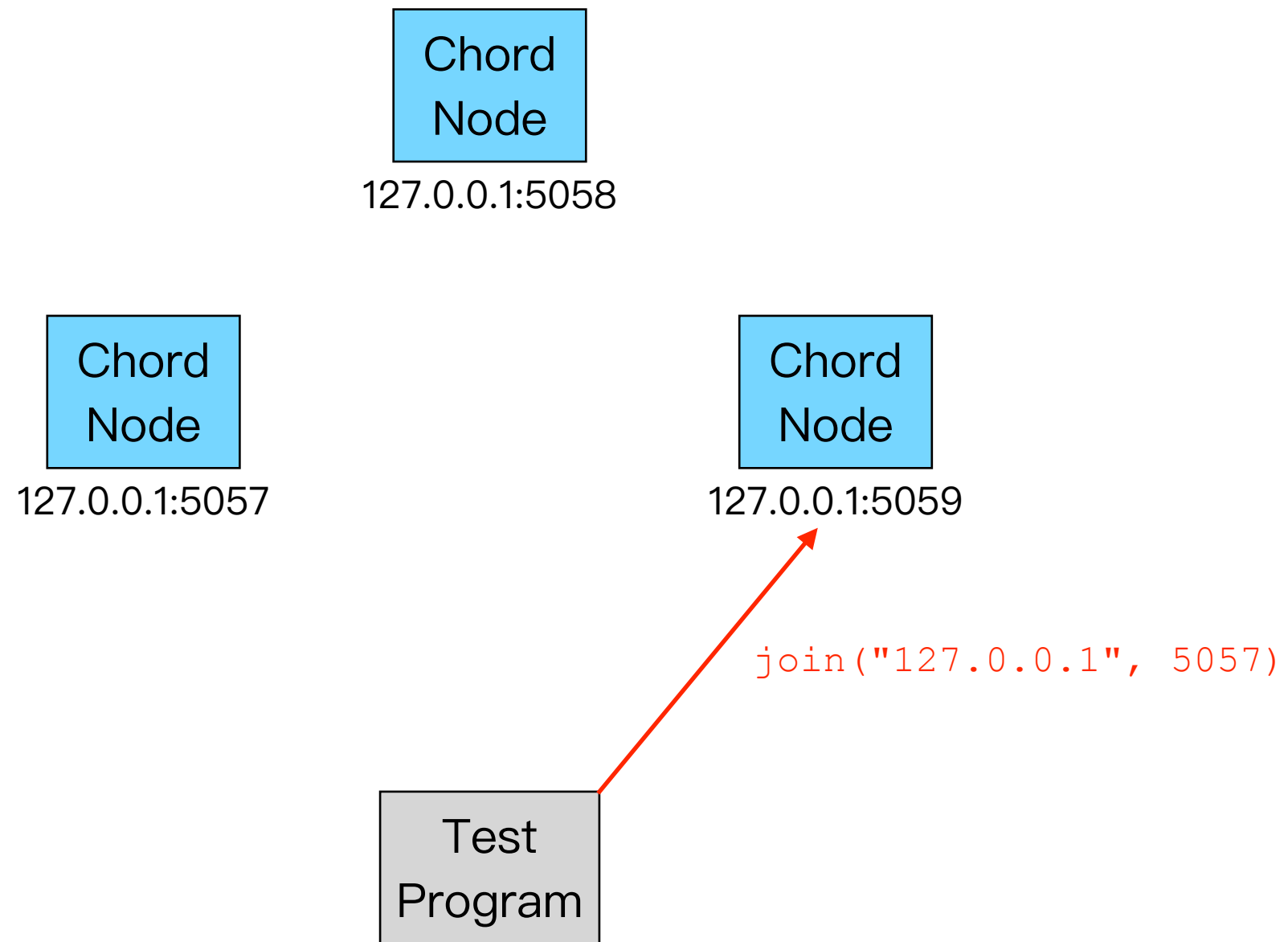
Chord System



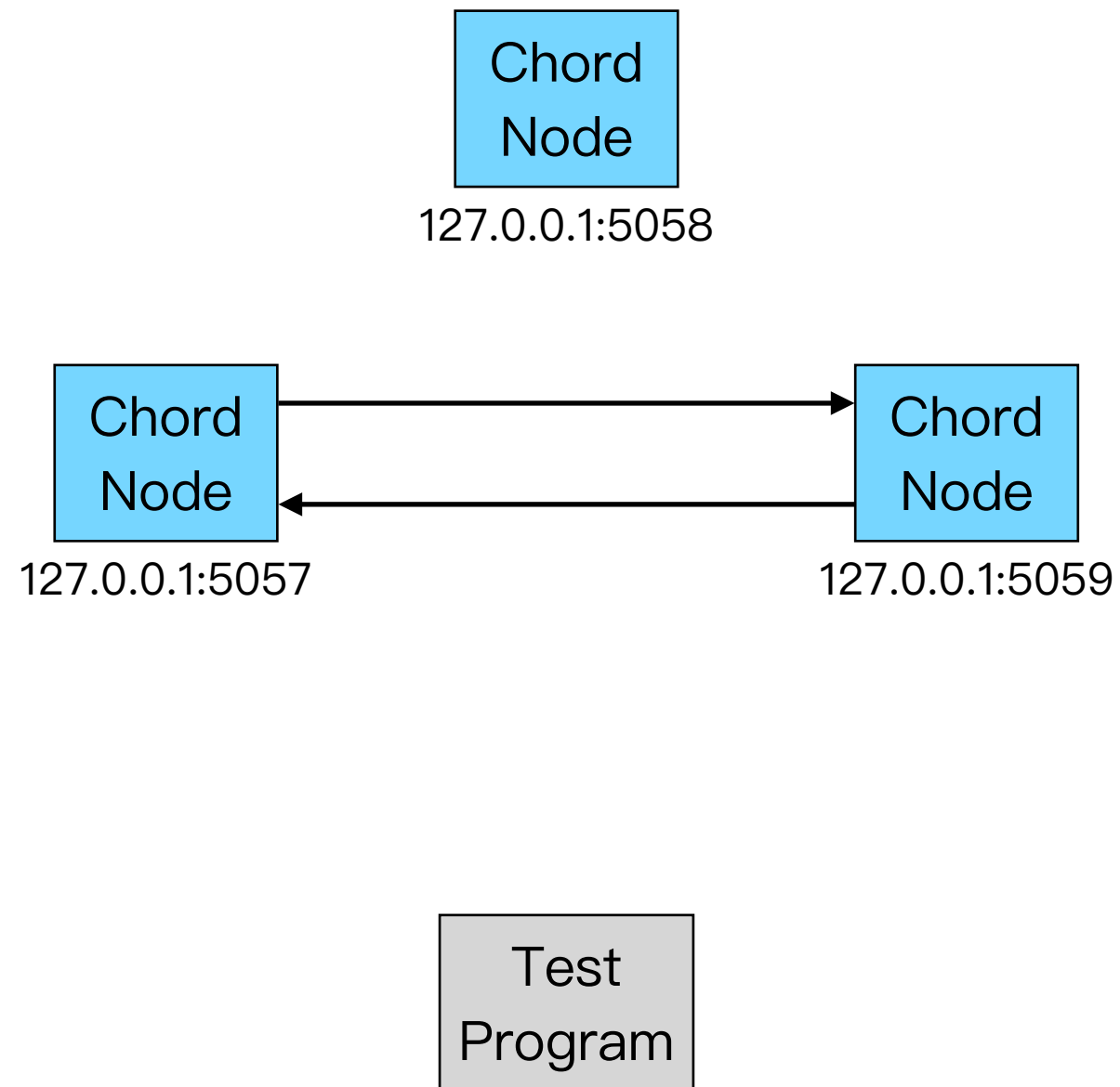
Chord System



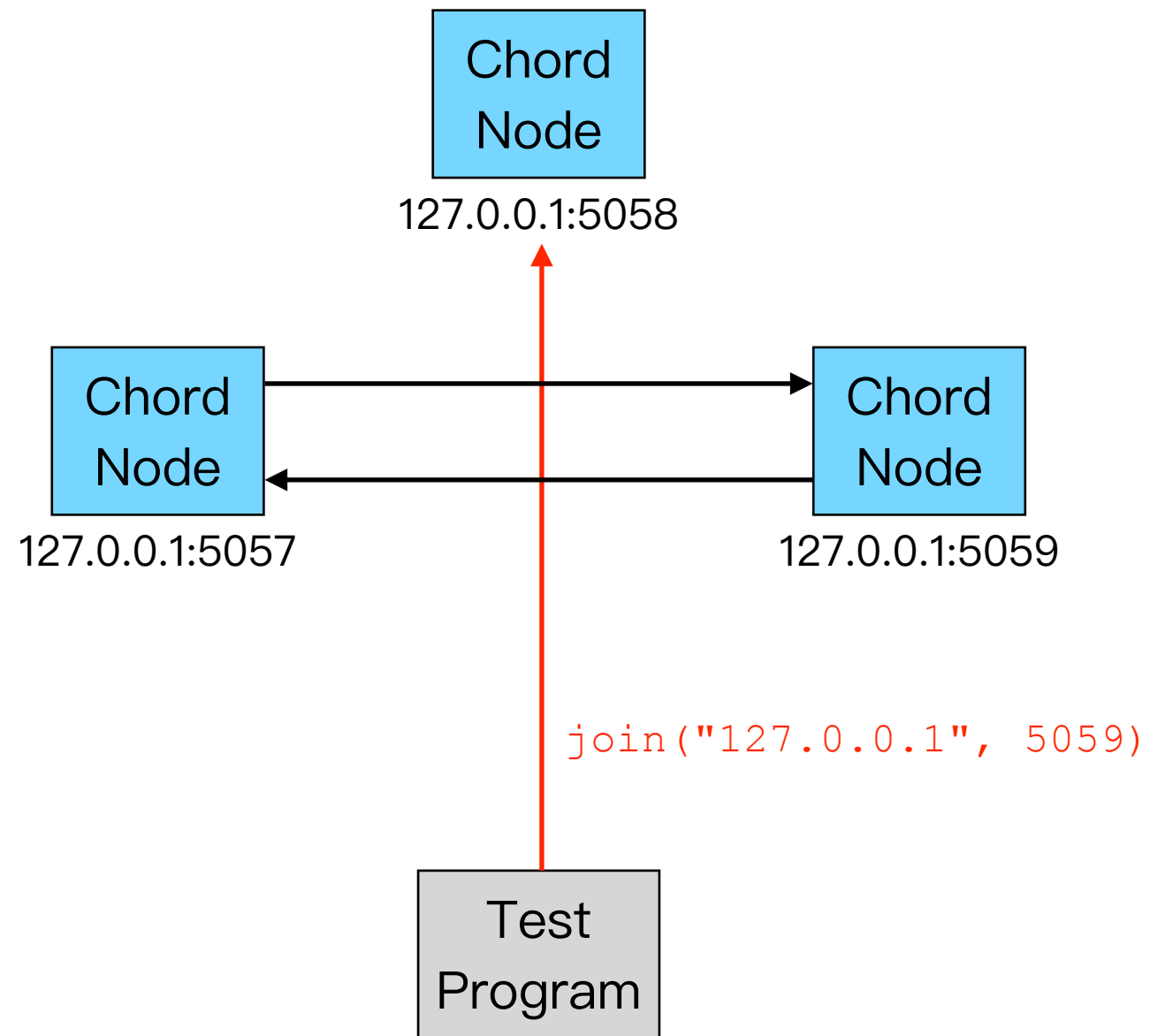
Chord System



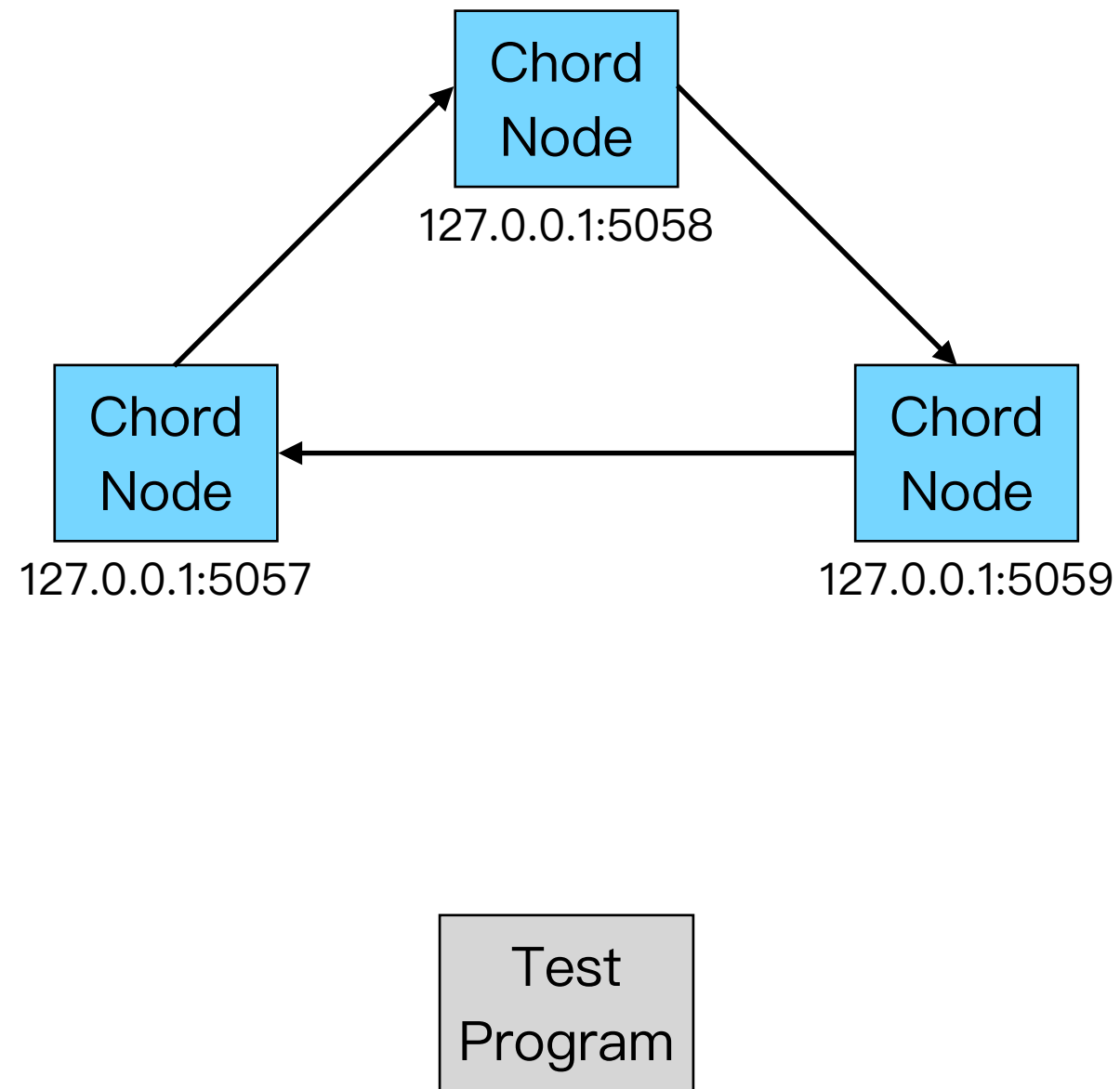
Chord System



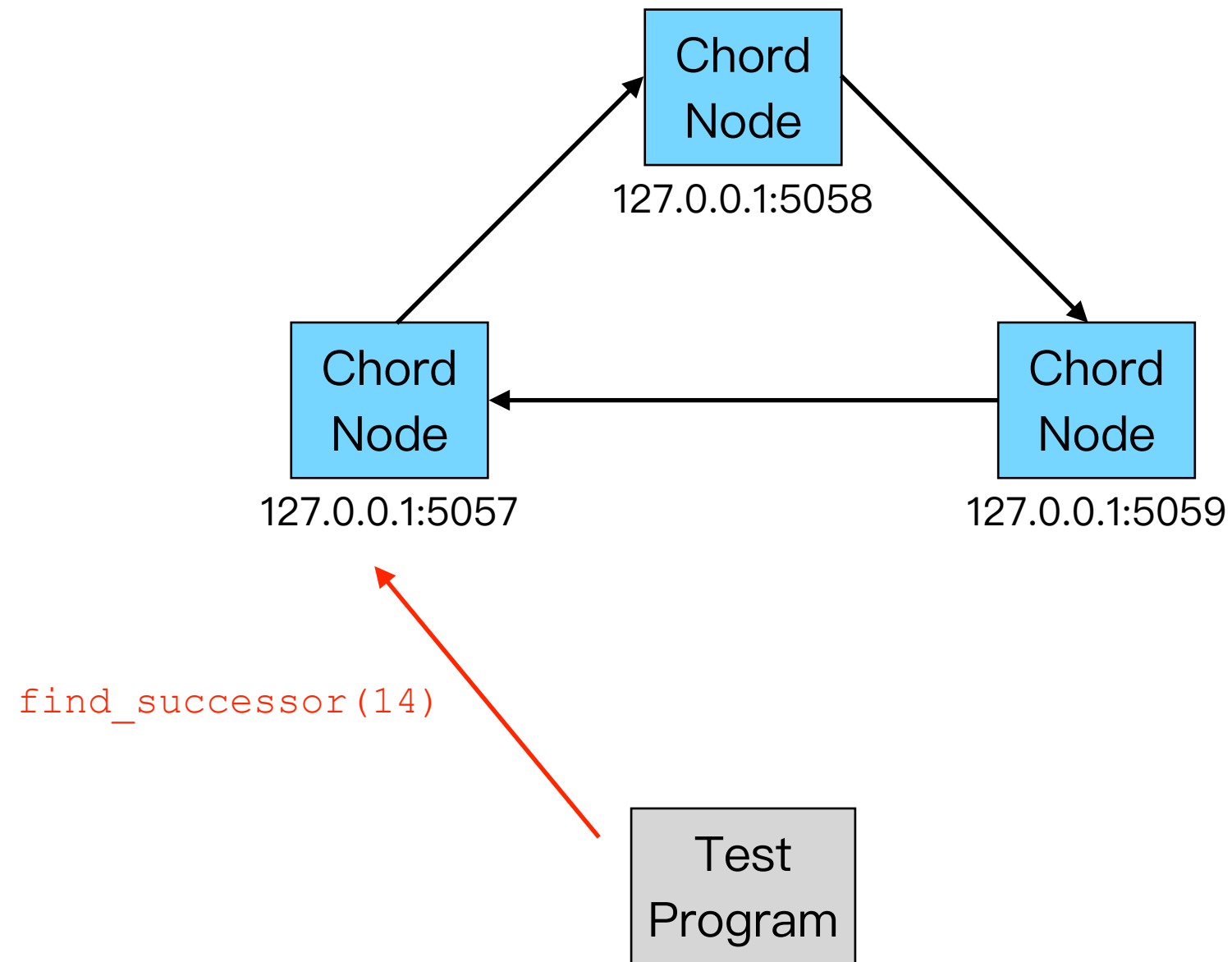
Chord System



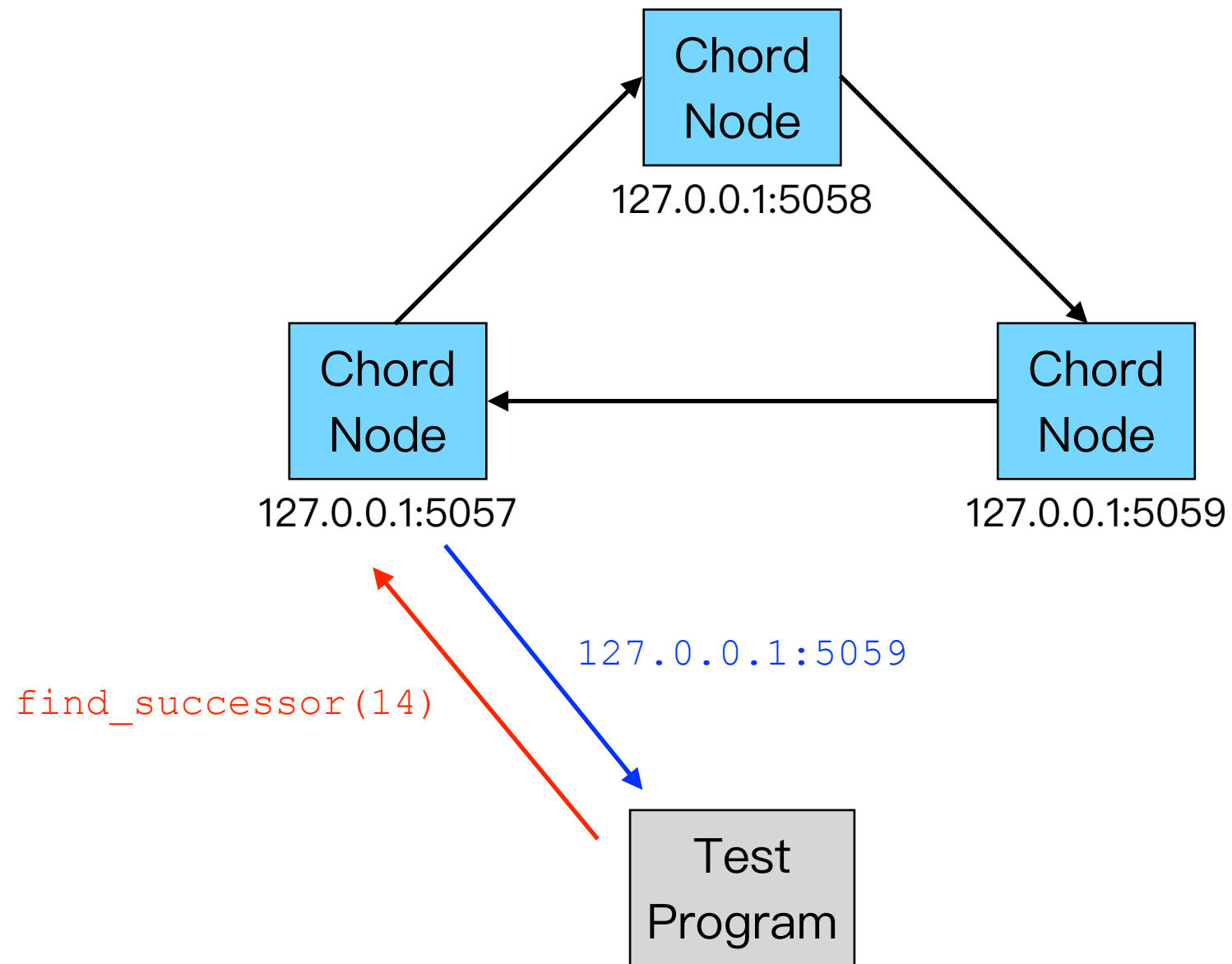
Chord System



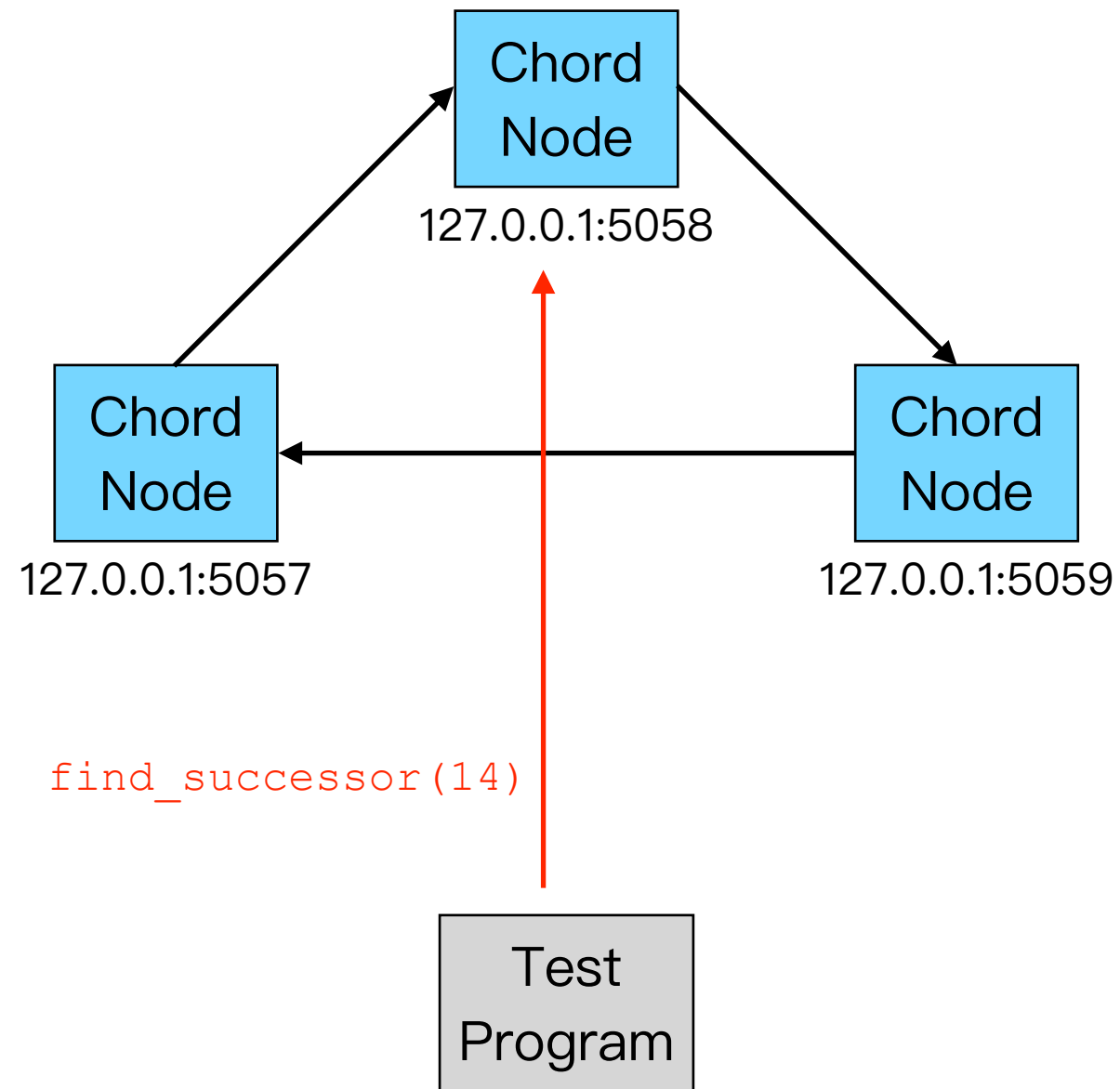
Chord System



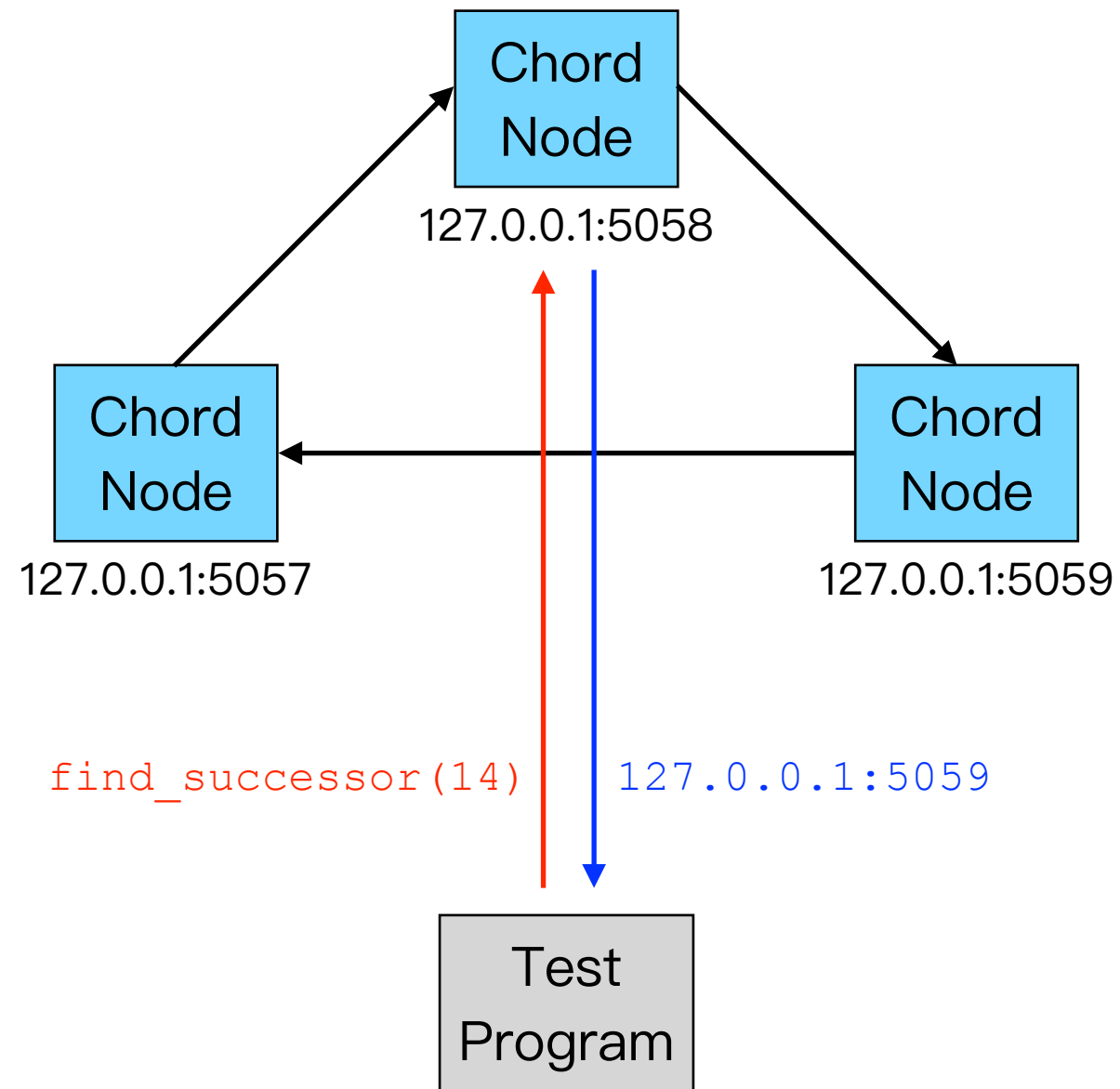
Chord System



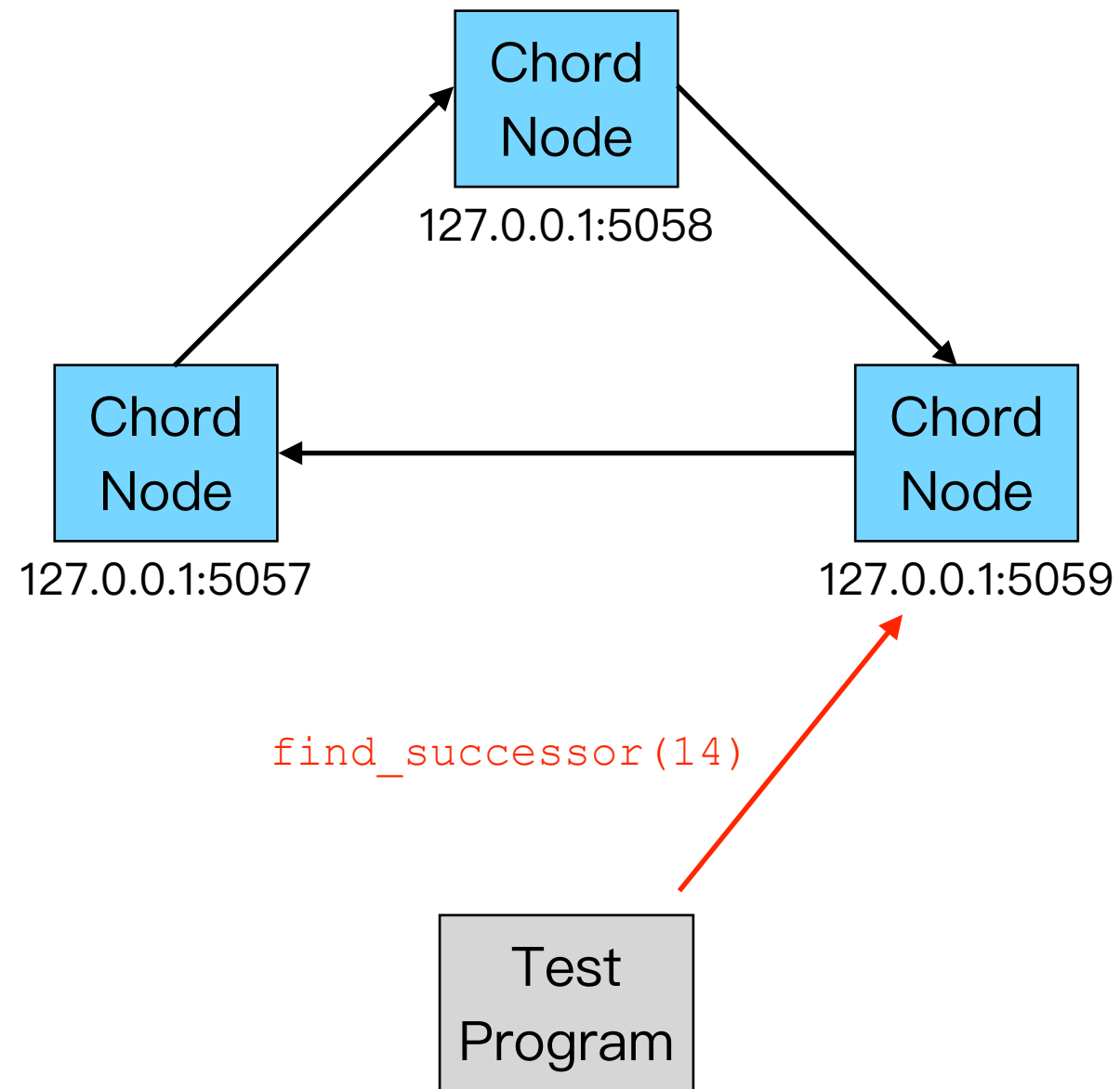
Chord System



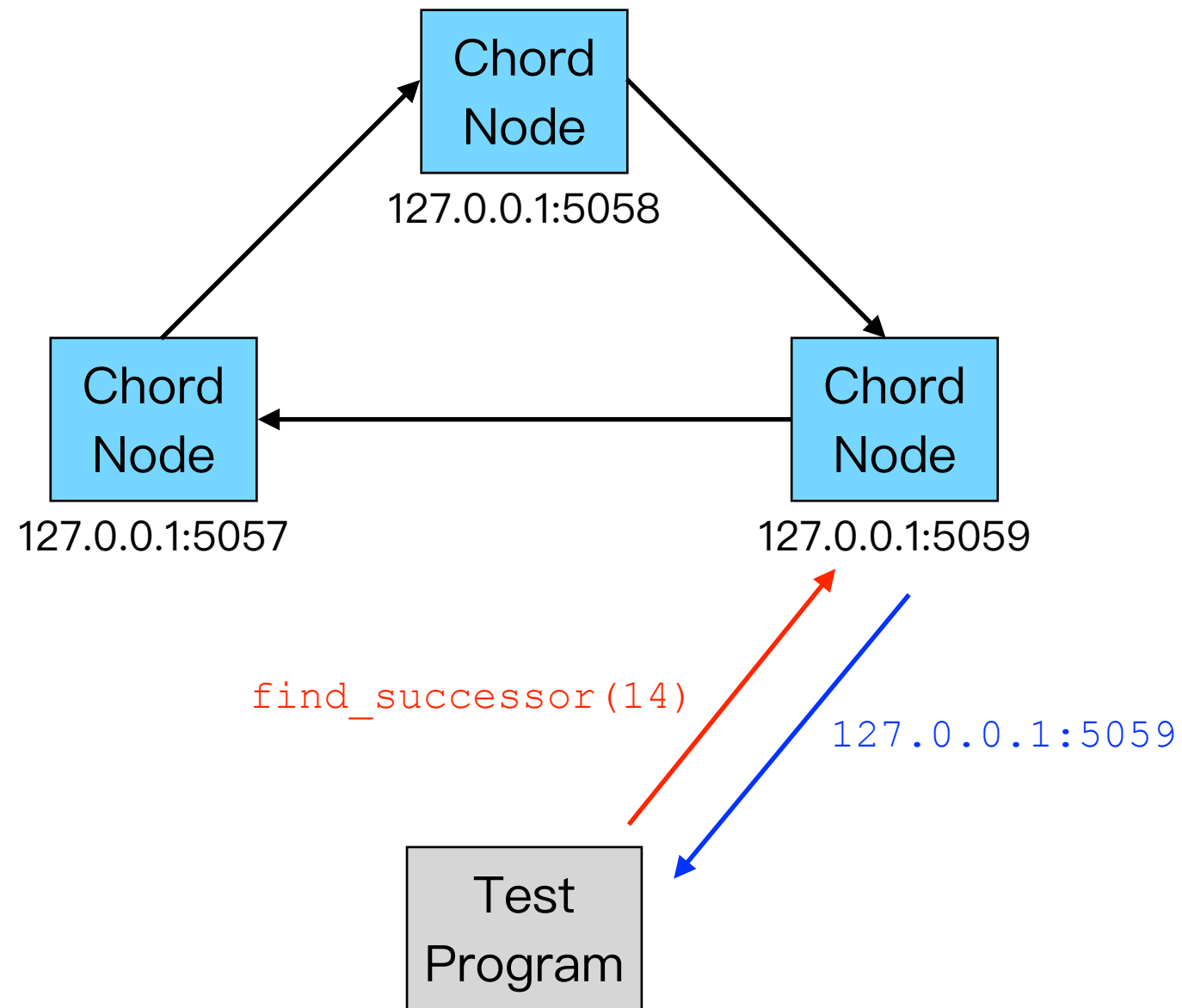
Chord System



Chord System



Chord System



Chord System

```
// Implementation (C++)
```

```
Node get_info() { return self; }
```

```
void create() {  
    predecessor.ip = "";  
    successor = self;  
}
```

```
void join(Node n) {  
    predecessor.ip = "";  
    rpc::client client(n.ip, n.port);  
    successor = client.call("find_successor", self.id).as<Node>();  
}
```

```
# Test script (Python)
```

```
client_1 = new_client("127.0.0.1", 5057)
```

```
client_2 = new_client("127.0.0.1", 5058)
```

```
print(client_1.call("get_info"))
```

```
print(client_2.call("get_info"))
```


```
client_1.call("create")
```

```
client_2.call("join", client_1.call("get_info"))
```

Grading Policy

- Correctness: 40%
- Message Complexity: 30%
- Fault Tolerance: 30%

Correctness (40%)

- We will test your Chord system by calling the "get_info", "create", "join", and "find_successor" RPCs

- Your Chord system must always return the correct answer when the "find_successor" RPC is called

Correctness (40%)

```
client_0.call("create")

client_2.call("join", client_0.call("get_info"))
client_1.call("join", client_2.call("get_info"))
client_4.call("join", client_1.call("get_info"))
client_5.call("join", client_0.call("get_info"))
client_3.call("join", client_4.call("get_info"))
# ...

ans = client_0.call("find_successor", 1234567)
# check ans
ans = client_1.call("find_successor", 1234567890)
# check ans
# ...
```

Message Complexity (30%)

- Message complexity when processing "find_successor" requests
- Message complexity when no external requests

Fault Tolerance (30%)

- We will test your Chord system by calling the "get_info", "create", "join", "find_successor" and "kill" RPCs
 - Killing one node at once (15%)
 - Killing two nodes at once (15%)

Grading Policy

- If your Chord system fails to meet **correctness**, you will not receive credit for the subsequent parts