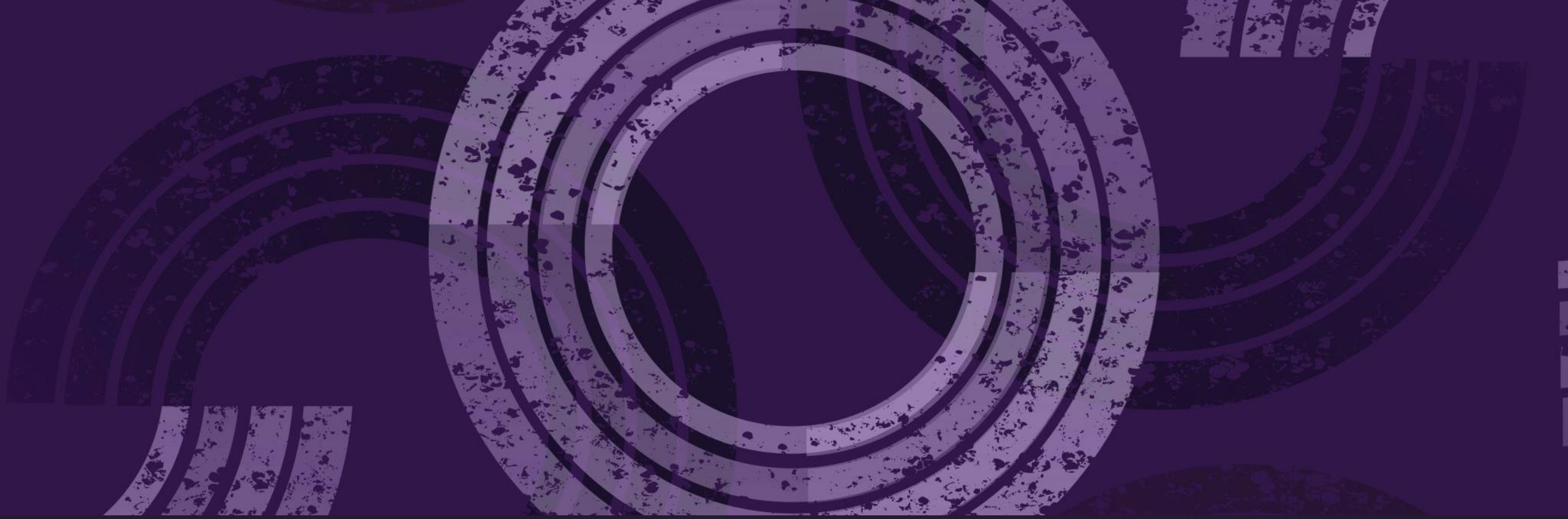
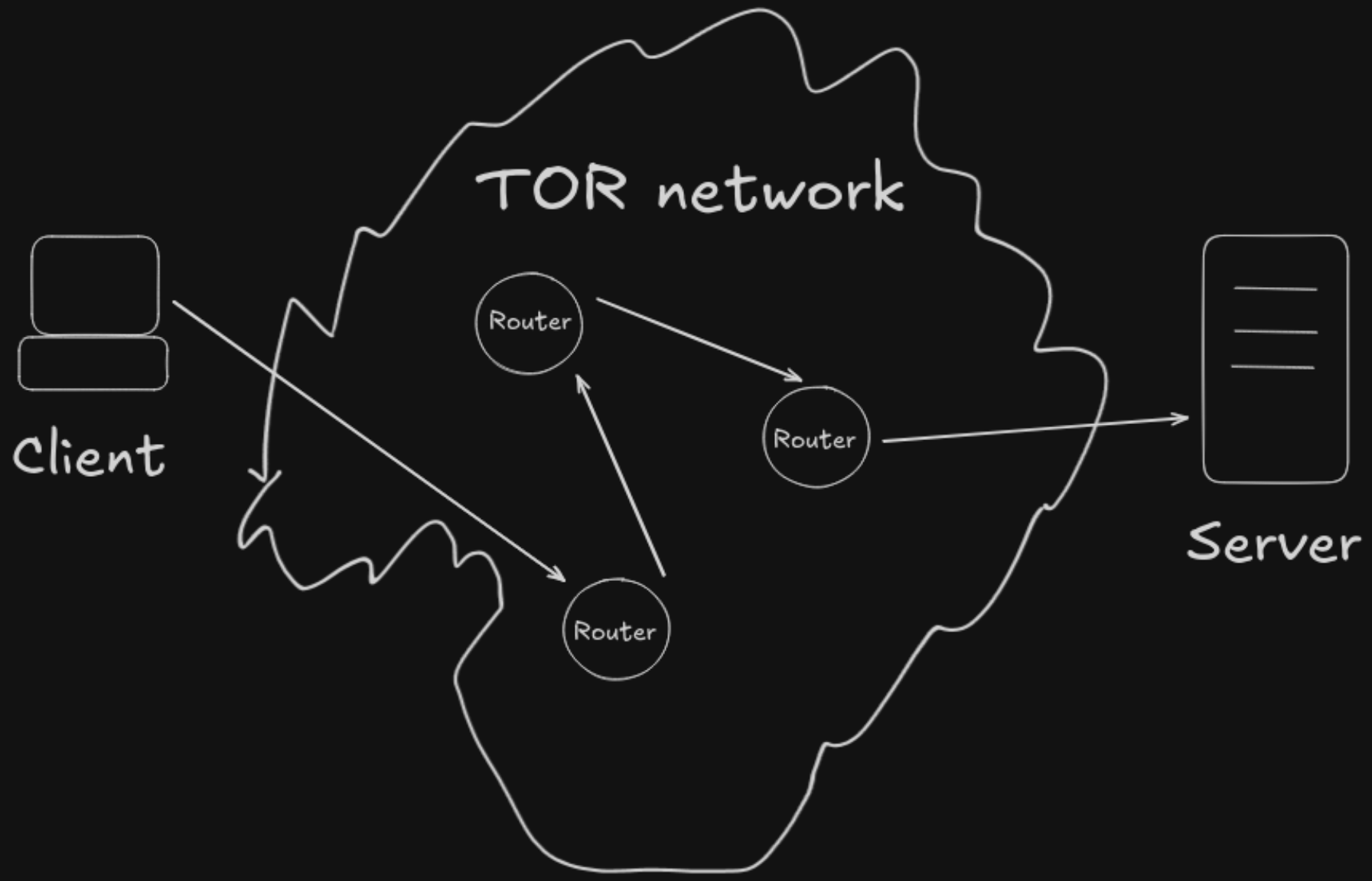
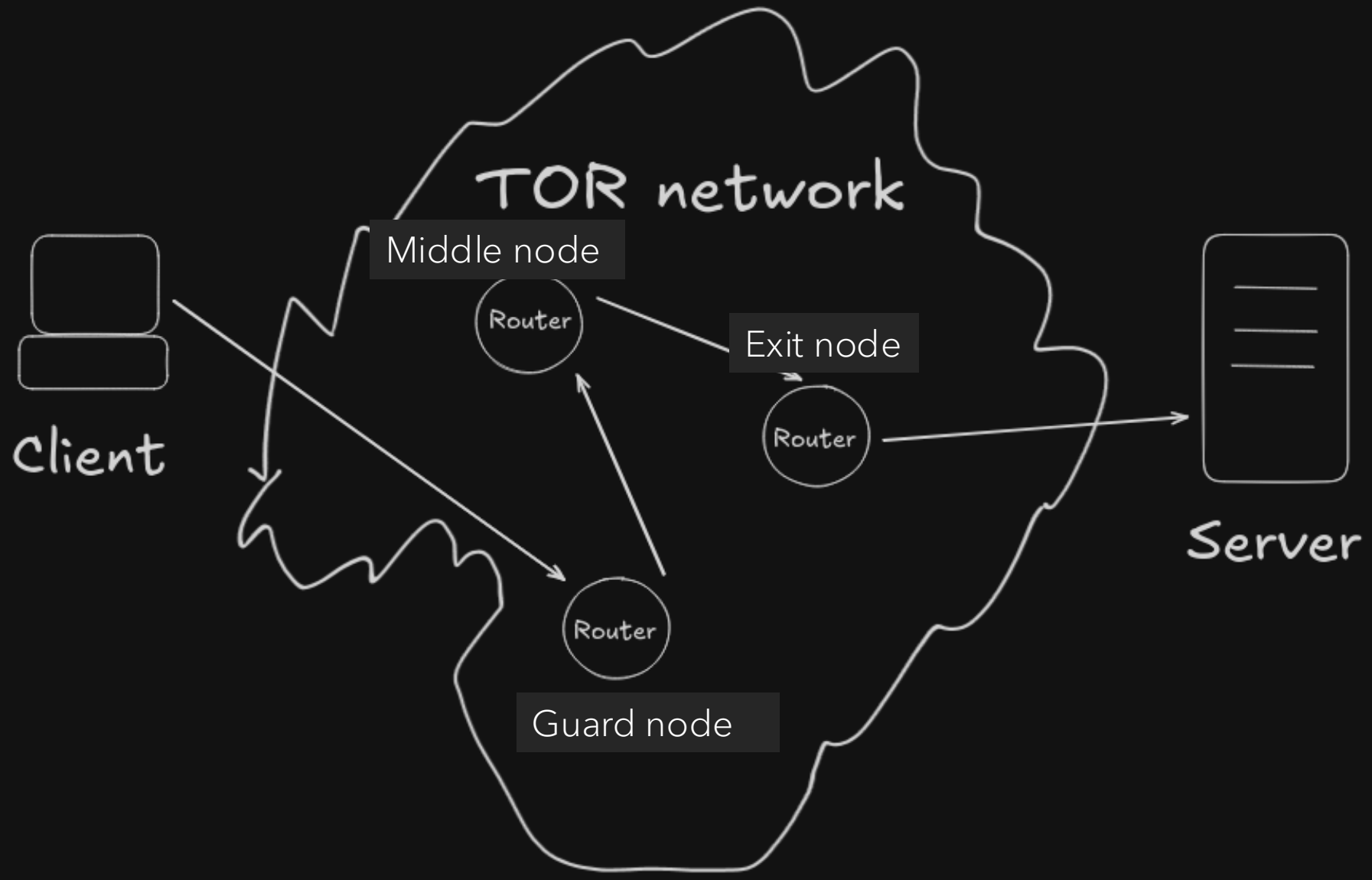


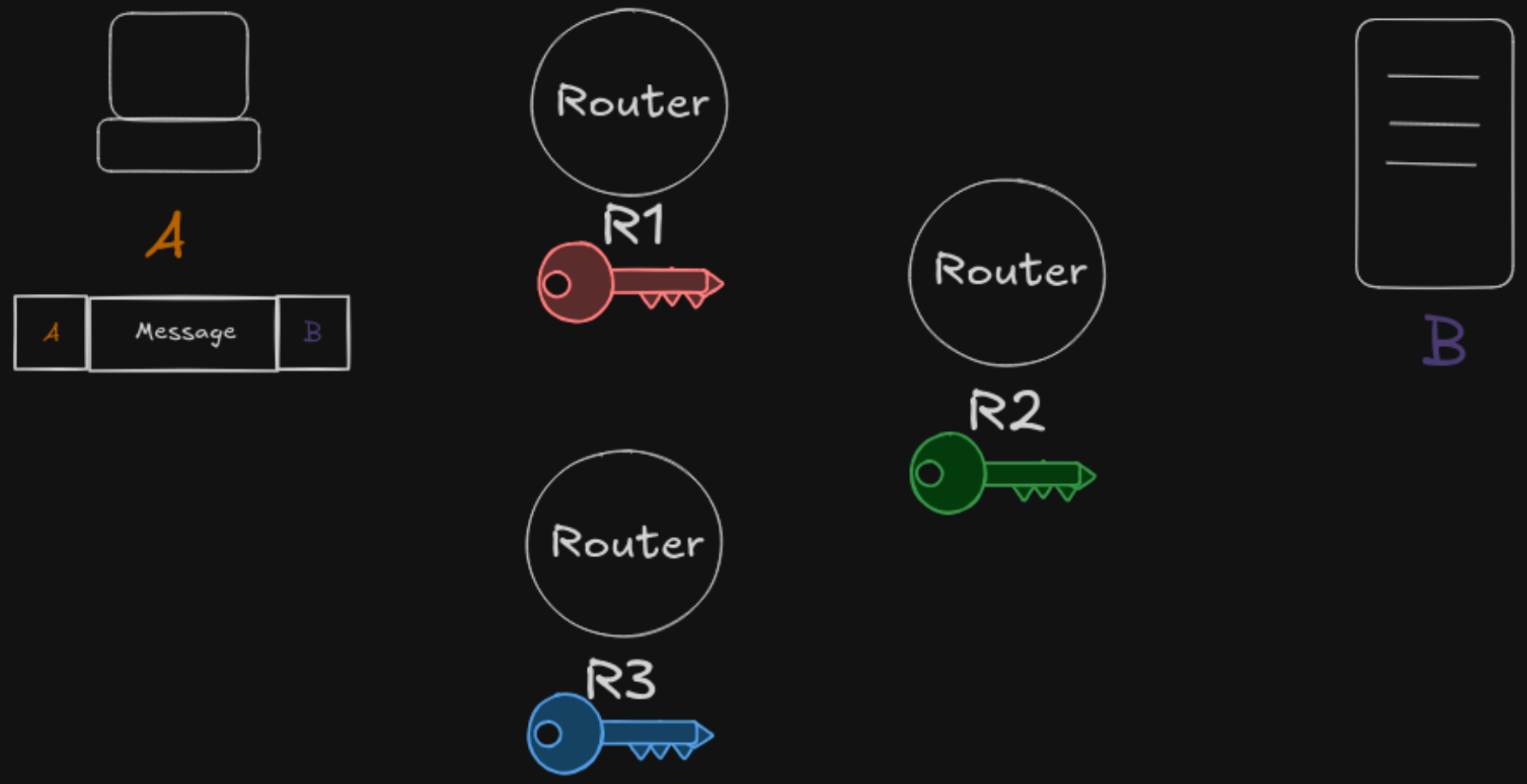
# TOR

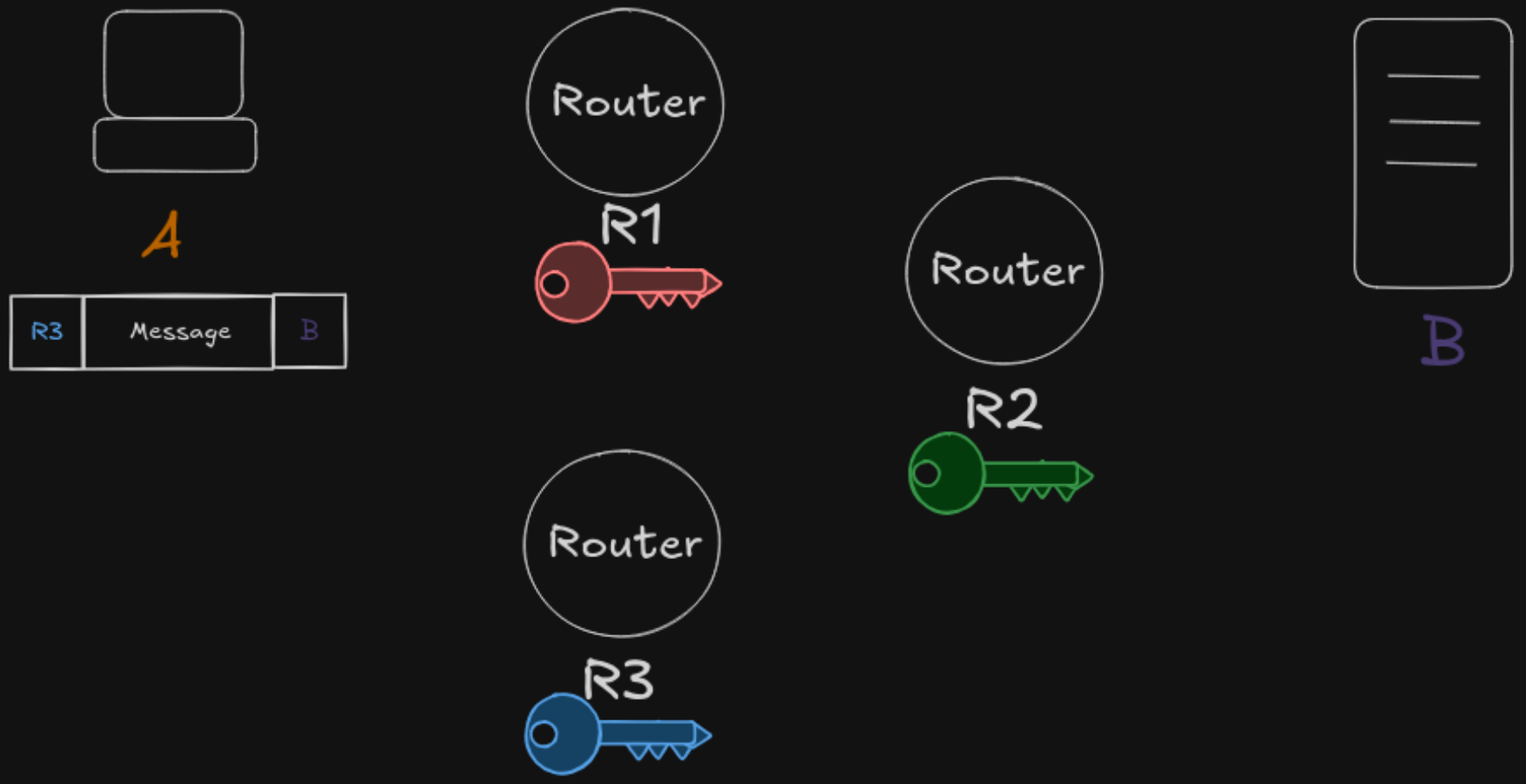
*Alex Babenko10044281,*
Tusshar Rana 100426359

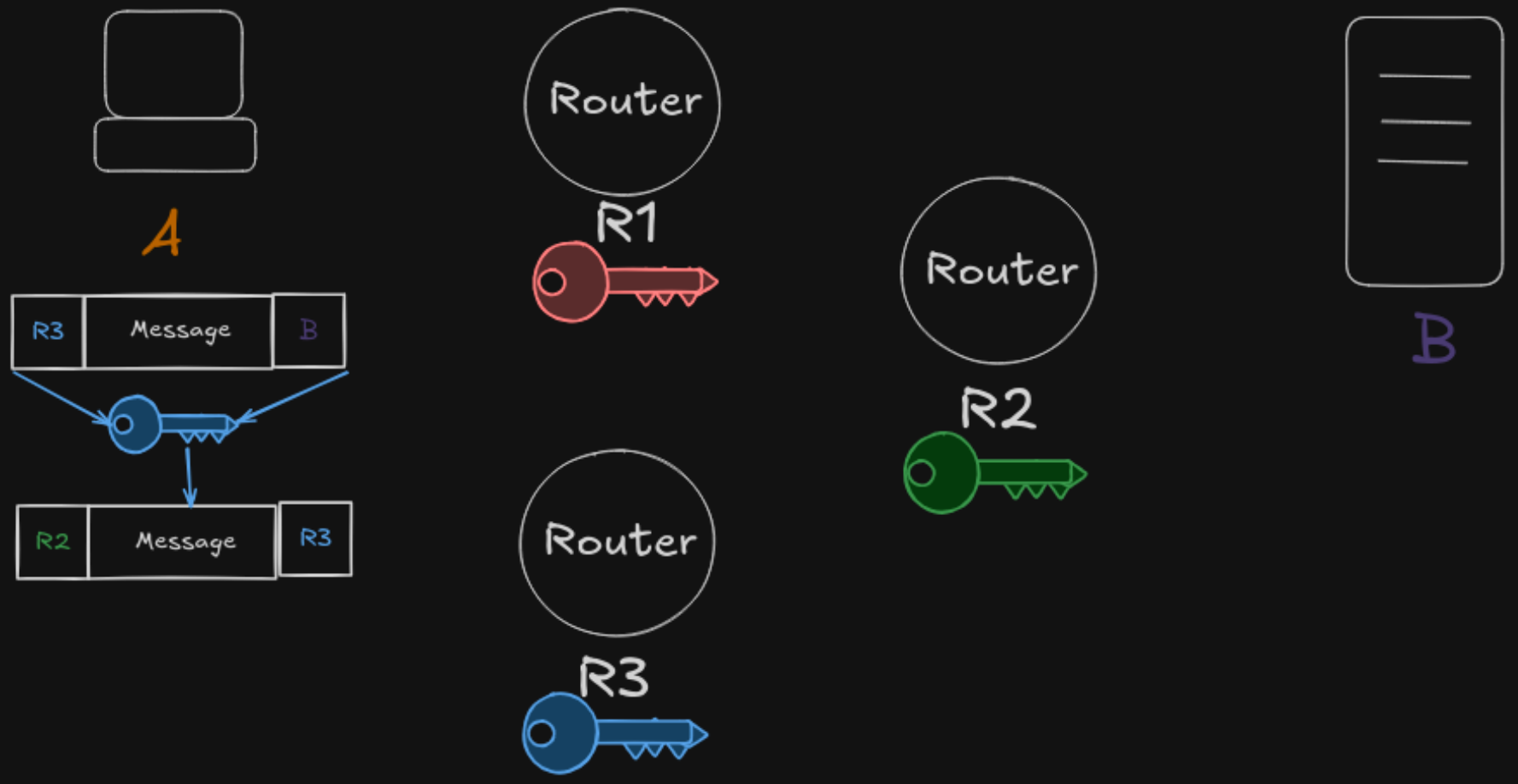

TOR network
Client
Router
Router
Router
Server

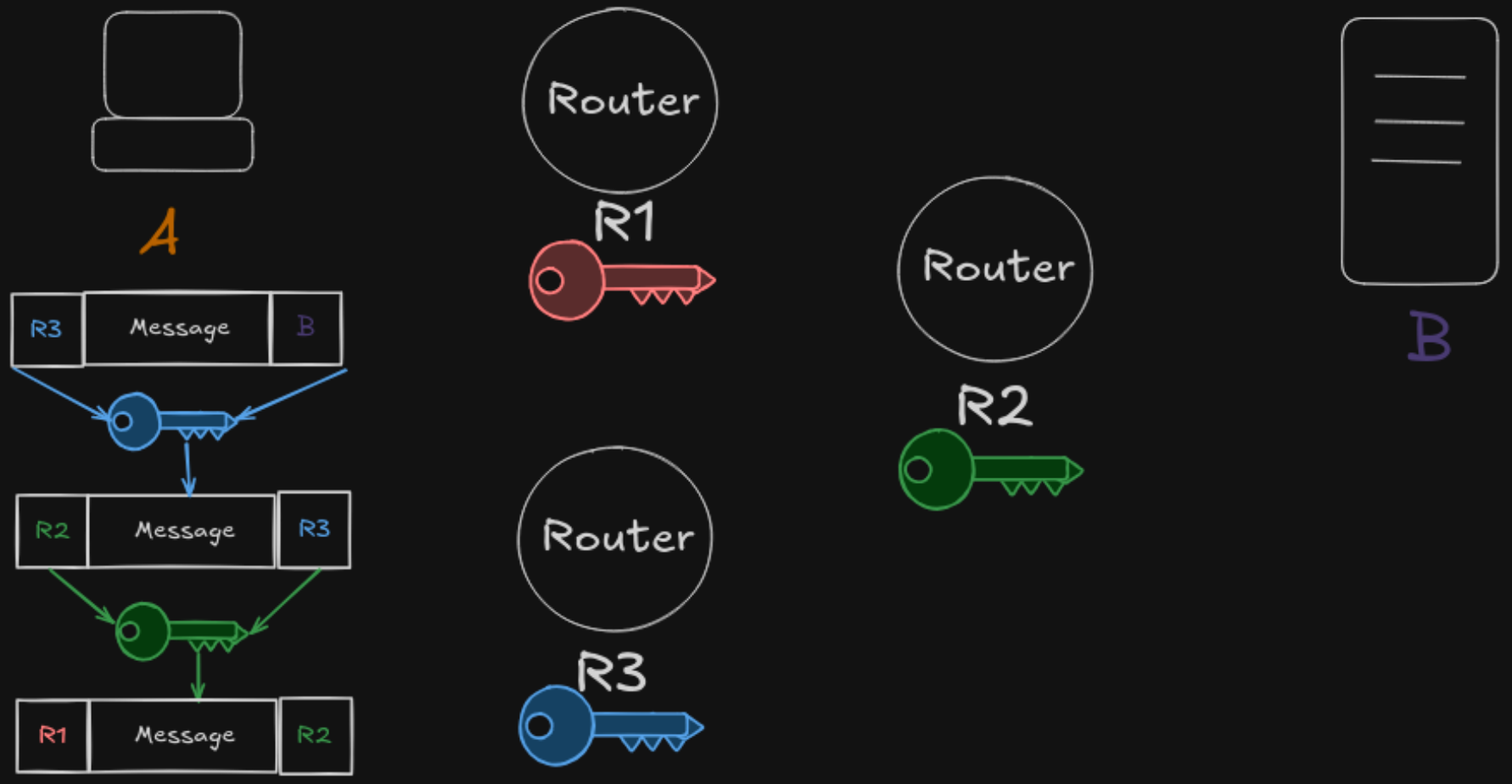

TOR network
Middle node
Router
Exit node
Router
Client
Server
Router
Guard node

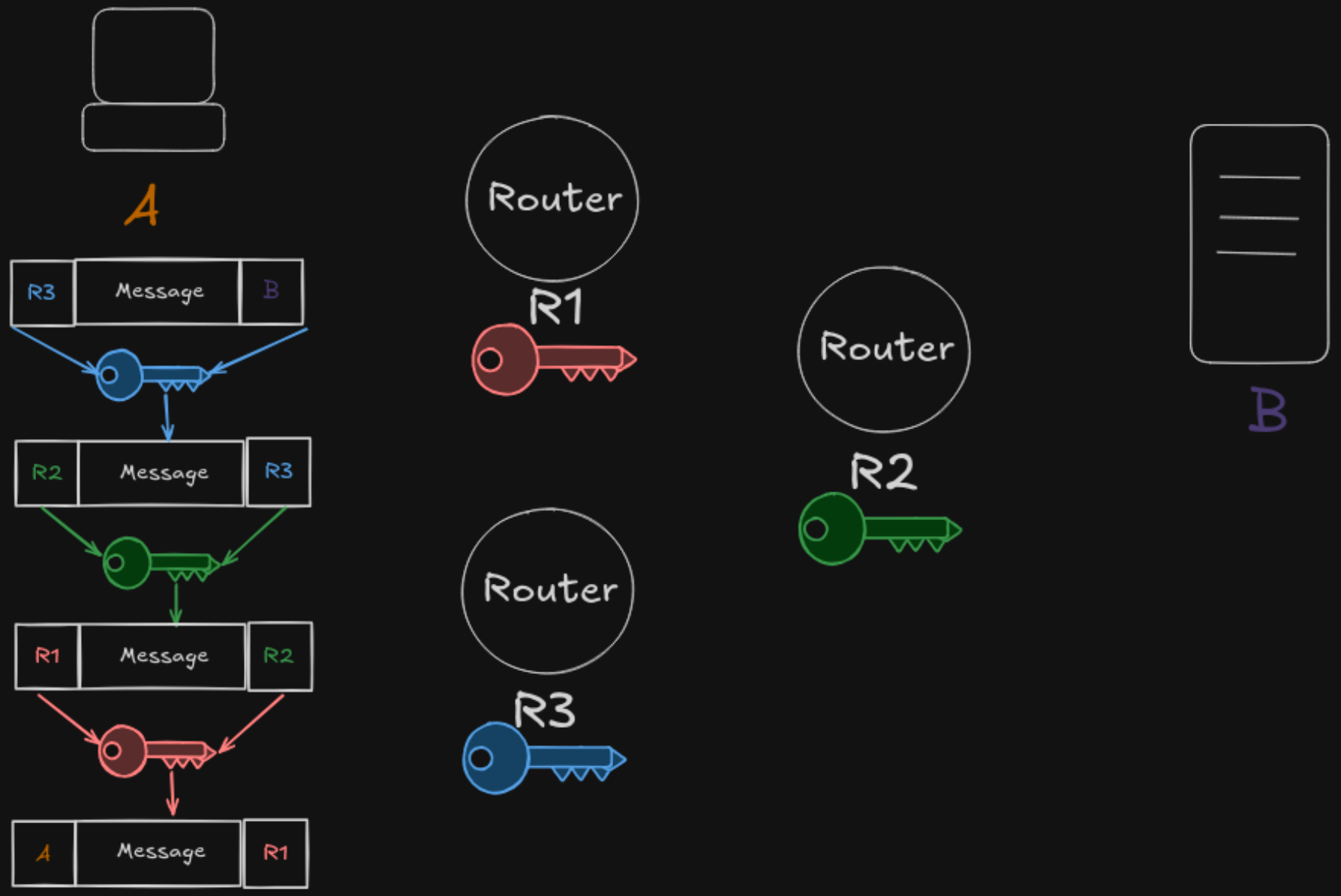
# BUT HOW CAN IT SEND PACKETS LIKE THAT?

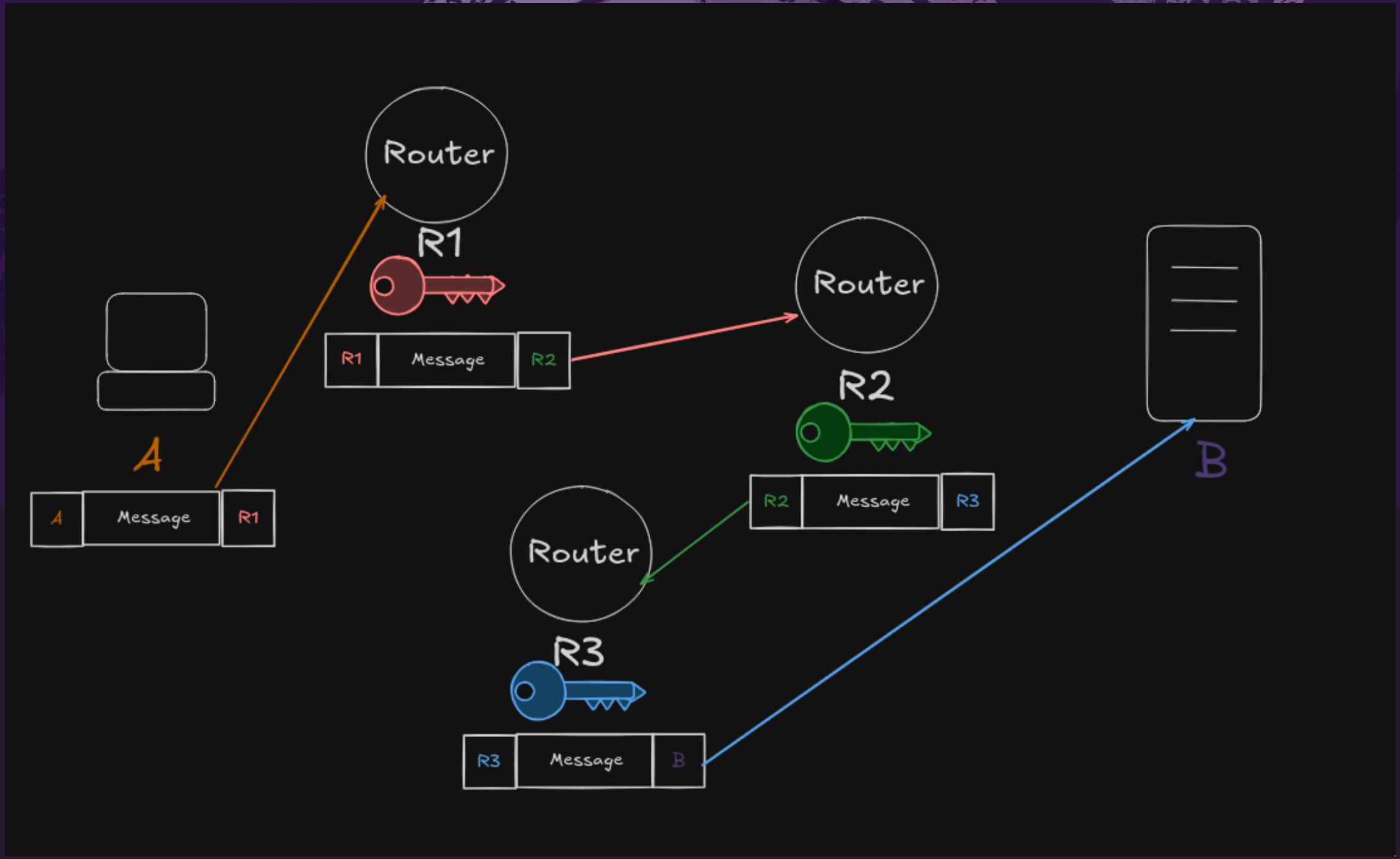

A
A
Message
B
Router
R1
Router
R2
Router
R3
B


A
R3
Message
B
Router
R1
Router
R2
Router
R3
B


A
R3
Message
B
R2
Message
R3
Router
R1
Router
R2
Router
R3
B


A
R3
Message
B
R2
Message
R3
R1
Message
R2
Router
R1
Router
R3
Router
R2
B


A
R3
Message
B
R2
Message
R3
R1
Message
R2
A
Message
R1
Router
R1
Router
R2
Router
R3
B


Router
R1
R1
Message
R2
Router
R2
R2
Message
R3
A
A
Message
R1
Router
R3
R3
Message
B
B

# BUT HOW DOES IT KNOW ABOUT THE ROUTERS?

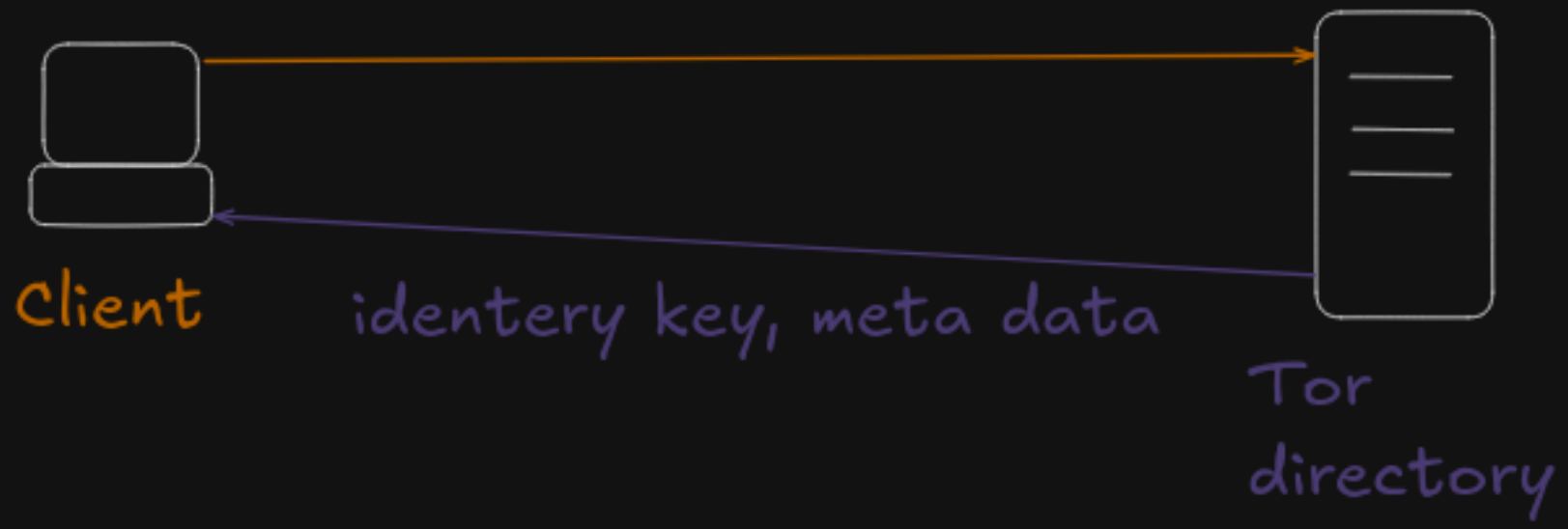
# TOR (On your computer)

Client

Trusted Tor directory list:

| IP, | identity hash, | onion key, | meta data |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |

1. **IDENTITY HASH** – HASH OF IDENTITY KEY AND META DATA
2. **IDENTIFY KEY** – IS A PERMINANT KEY THAT IS USED TO IDENTIFY A ROUTER, IT IS USED IN COMBINATION OF " TLS CERTIFICATES, ROUTER DESCRIPTOR ( **ONION KEYS** IT IS USING, ADDRESS, BANDWIDTH, EXIT POLICY, ETC)
3. **ONION KEYS** – ARE PUBLIC KEYS


Client
identery key, meta data
Tor
directory

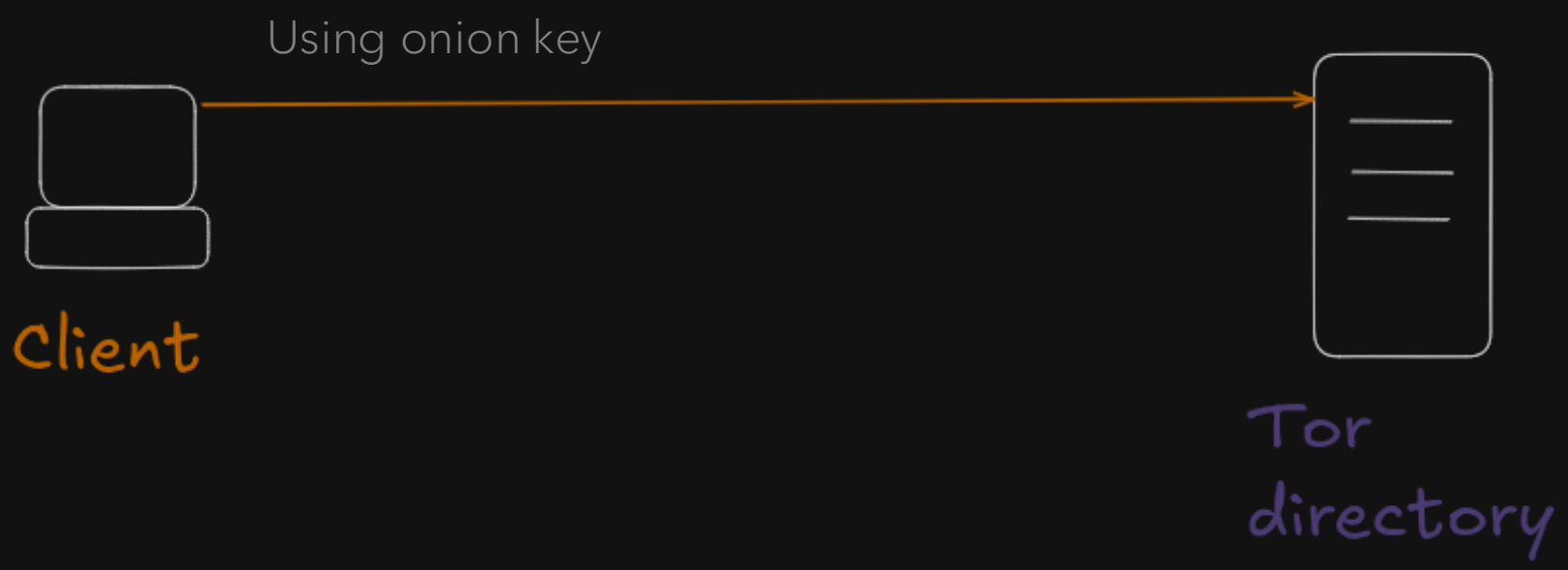
identery key, meta data

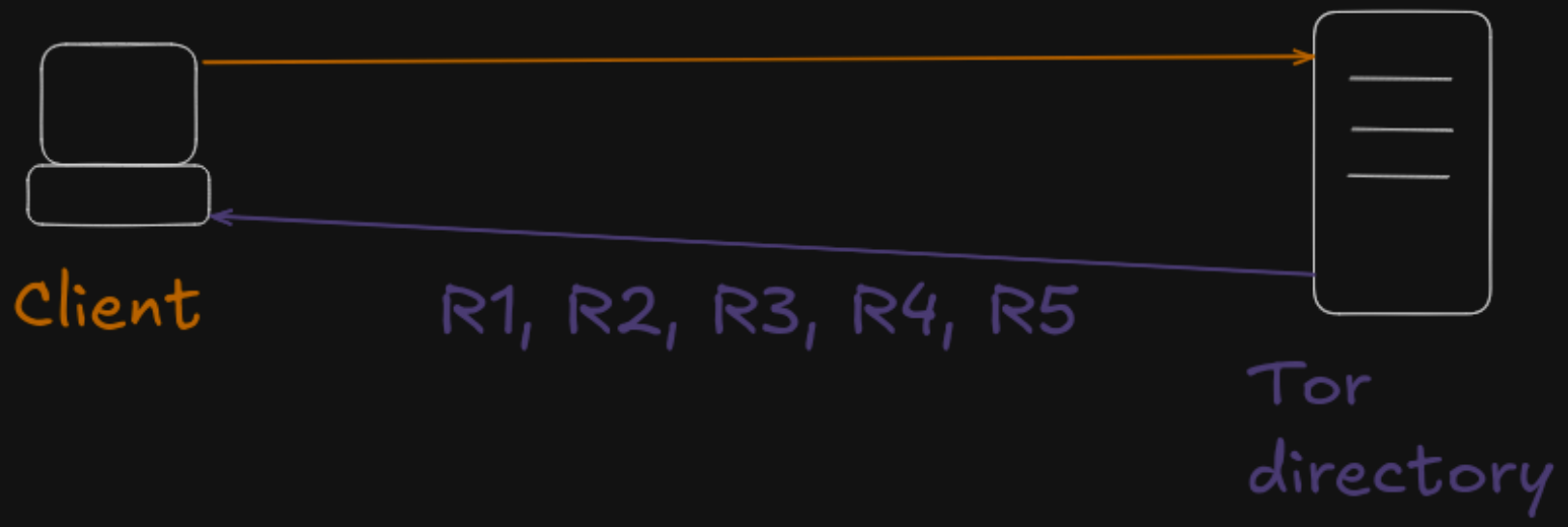
↓

Hash()

↓

Hash from Computer = Hash from Directory


Using onion key
Client
Tor
directory

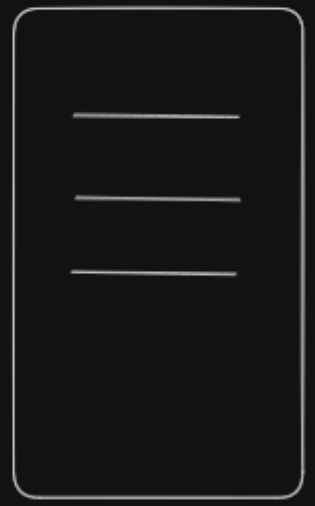

Client
R1, R2, R3, R4, R5
Tor
directory


Client
R1, R2, R3, R4, R5
Tor
directory

Client

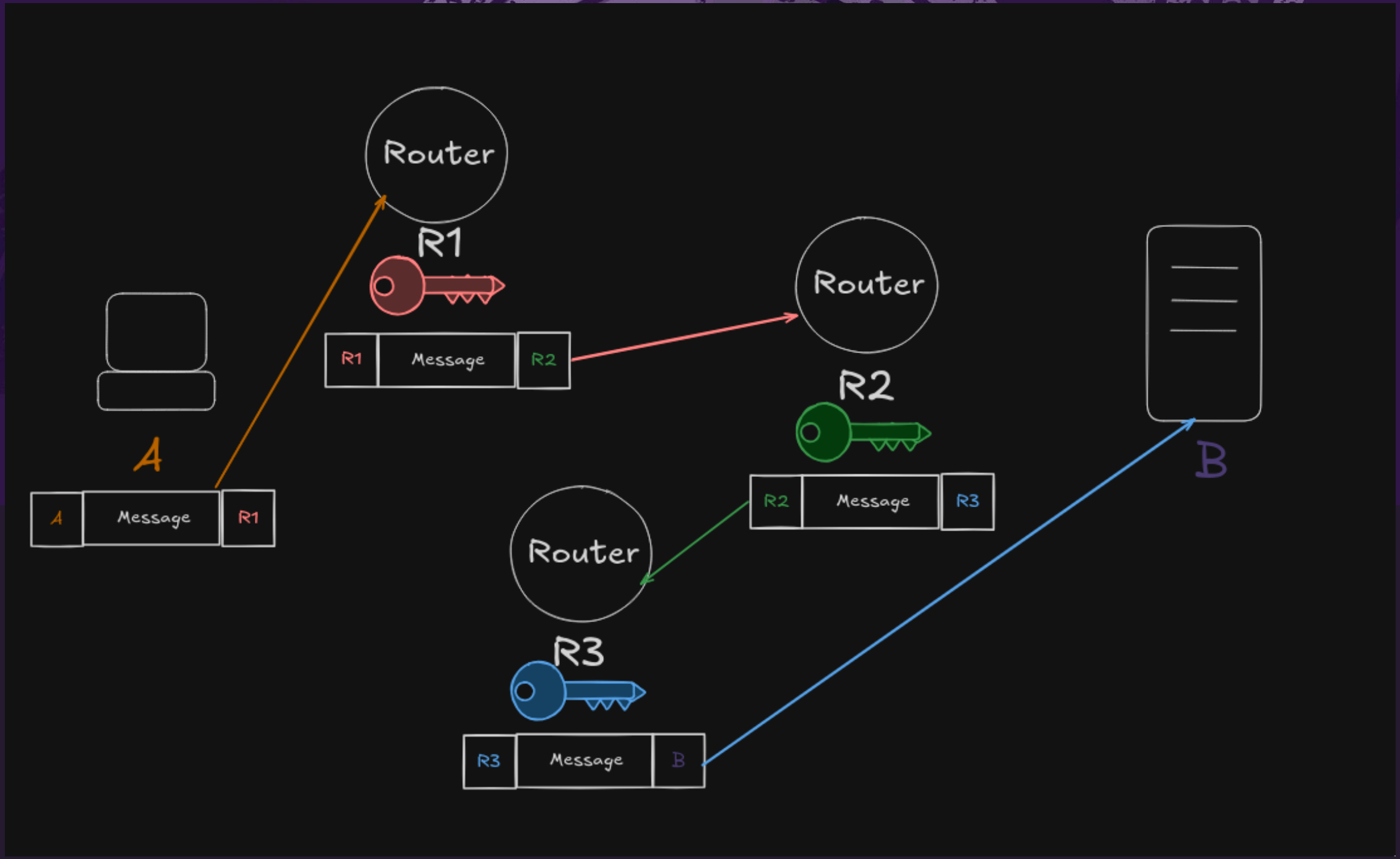
list of routers:

| IP, | Identity hash, | onion keys, | meta data |
|---|---|---|---|
| | | | |
| | | | |



Tor directory

# THE SAME CHECK WE DID WITH **DIRECTORY SERVER**(USING IDENTITY KEY) WILL HAPPEN WITH **ROUTERS** TOO

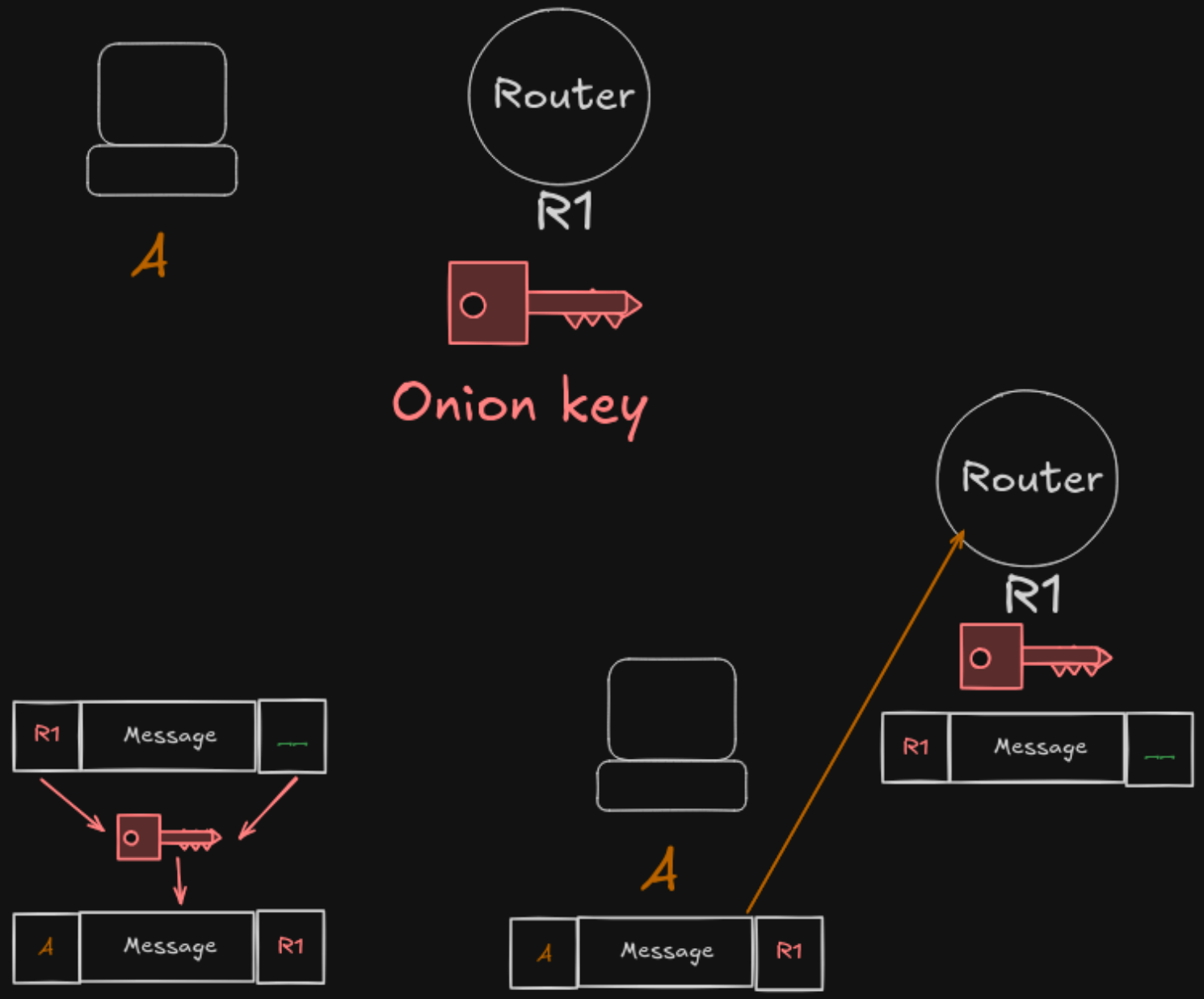

Router
R1
R1
Message
R2
Router
R2
R2
Message
R3
A
A
Message
R1
Router
R3
R3
Message
B
B

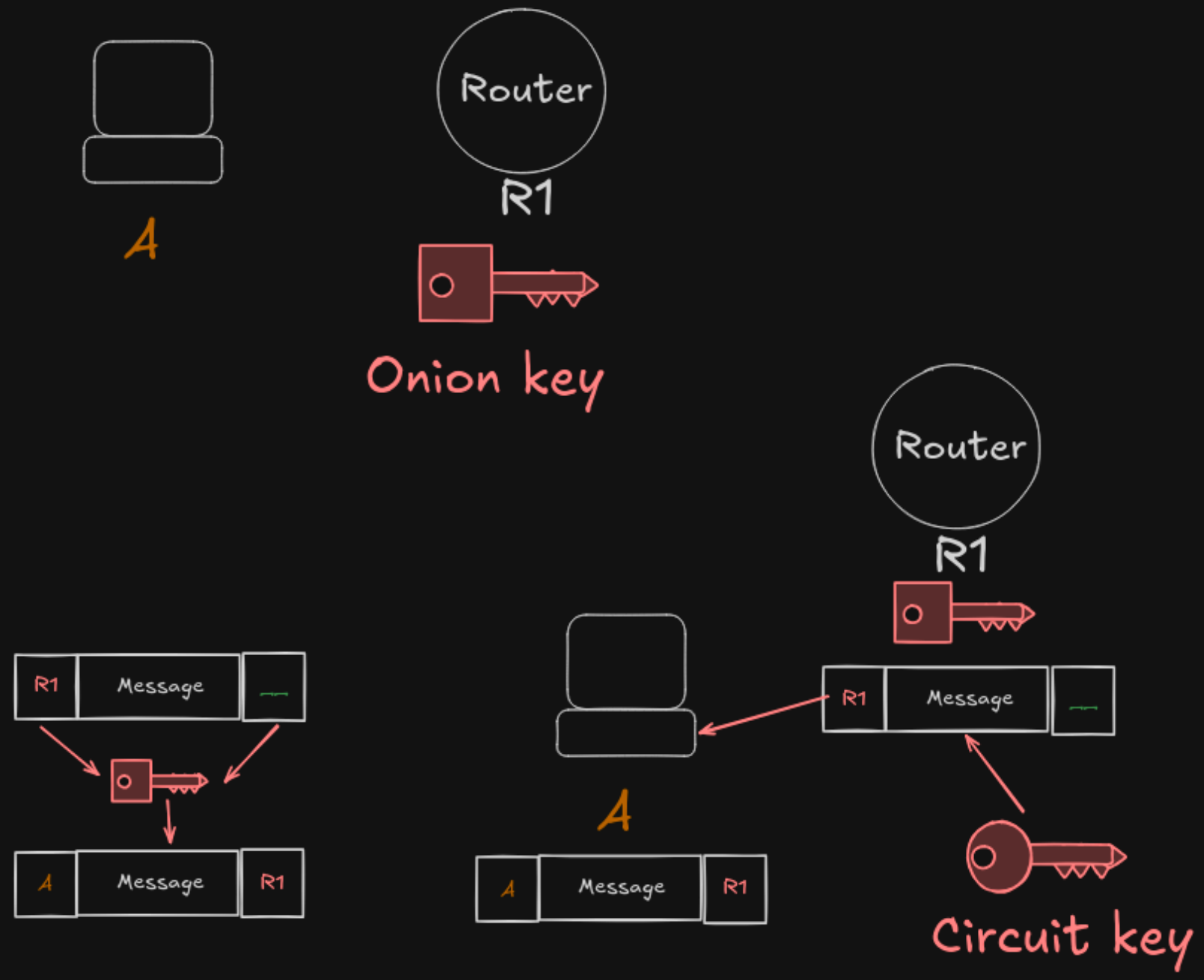
# BUT HOW DOES IT KNOW WHAT WILL BE A ROUTE(R1, R2, R3)?

# IT WILL LOOK AT THE META DATA AND LOCALLY DECIDE WHAT WILL BE THE BEST ROUTES

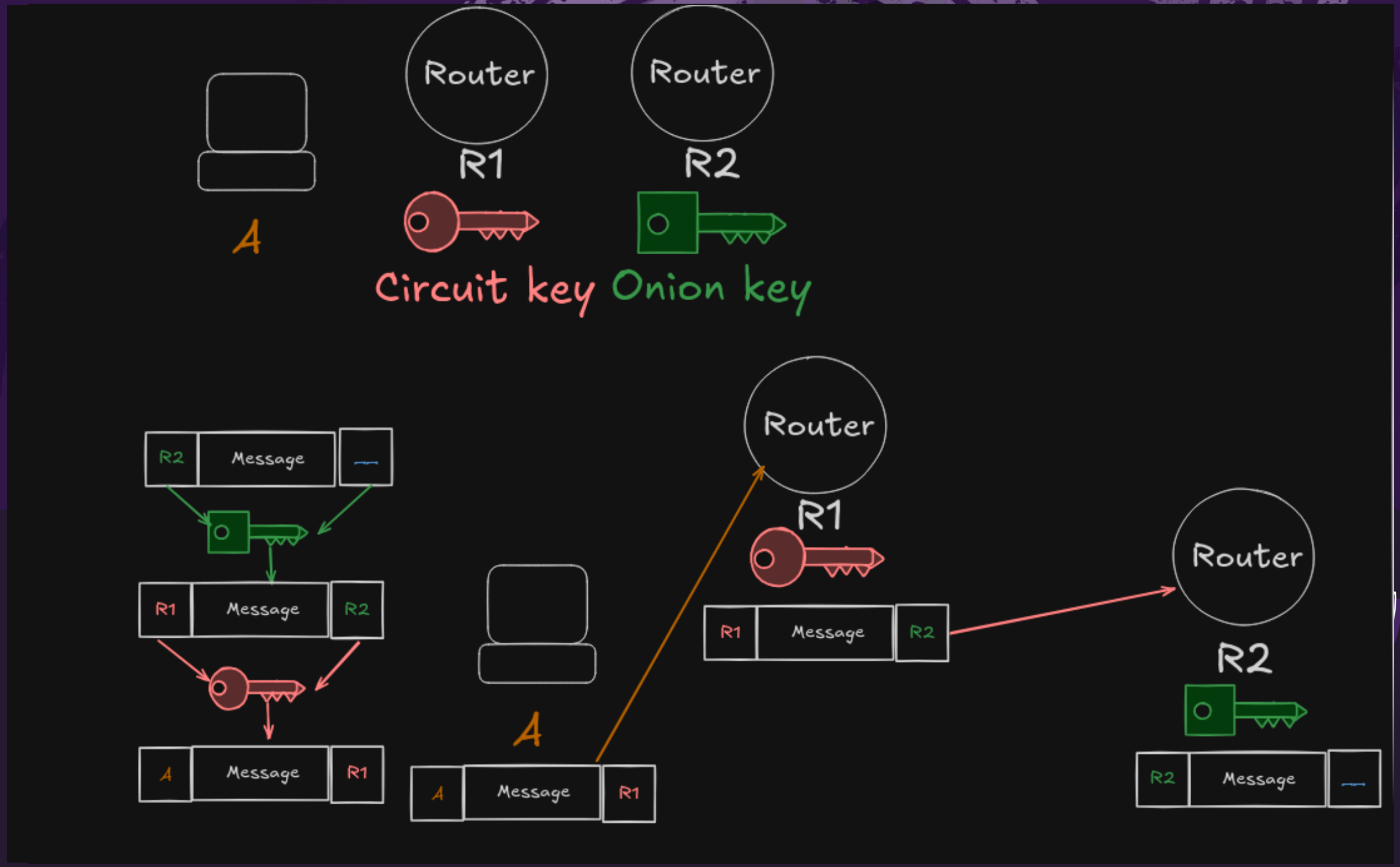
*Based on* location, speed, and bandwidth

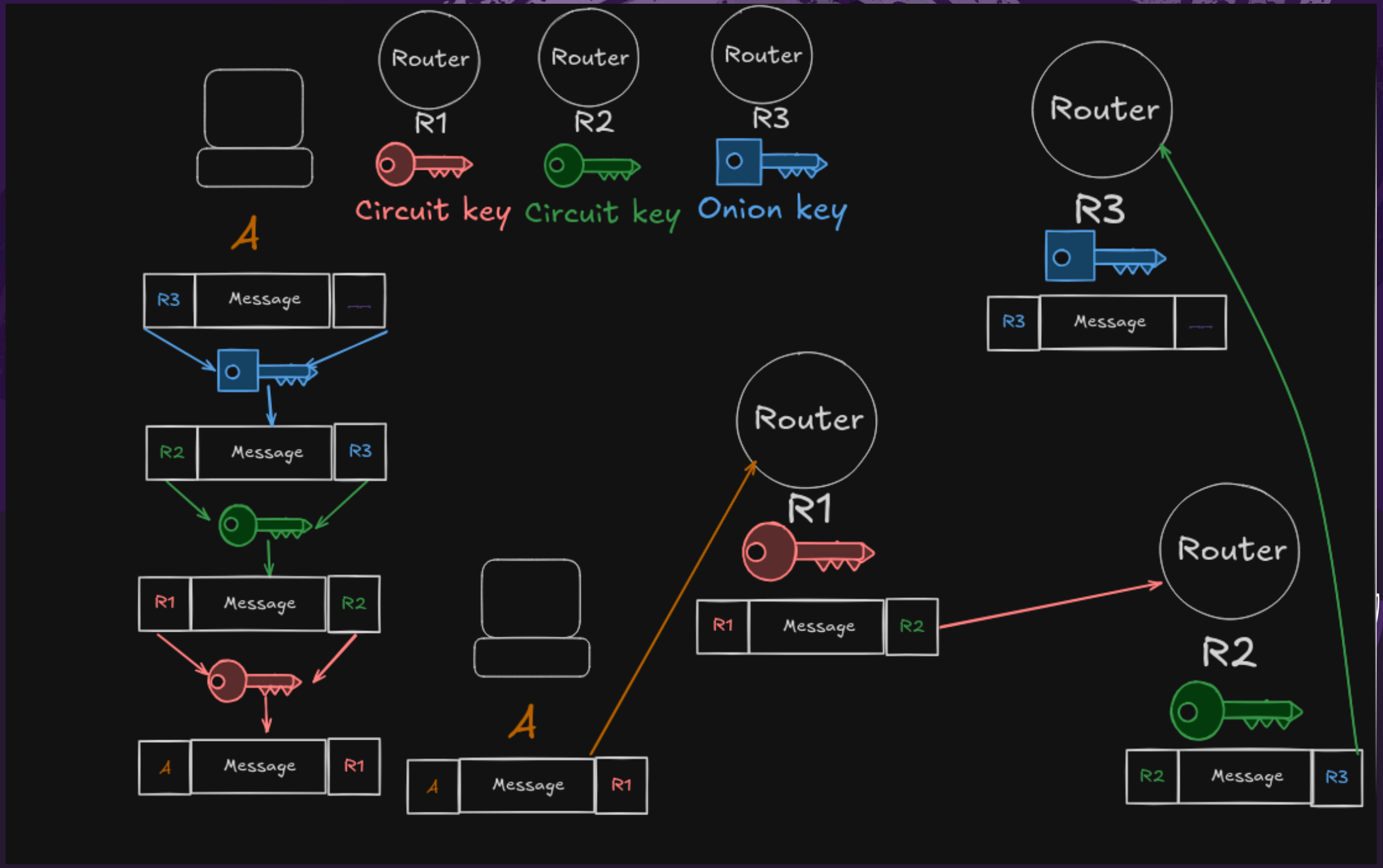
# BUT HOW THEY KNOW THIS ROUTE WILL WORK ON PRACTICE AND THAT EVERY NODE IS LEGITIMATE?

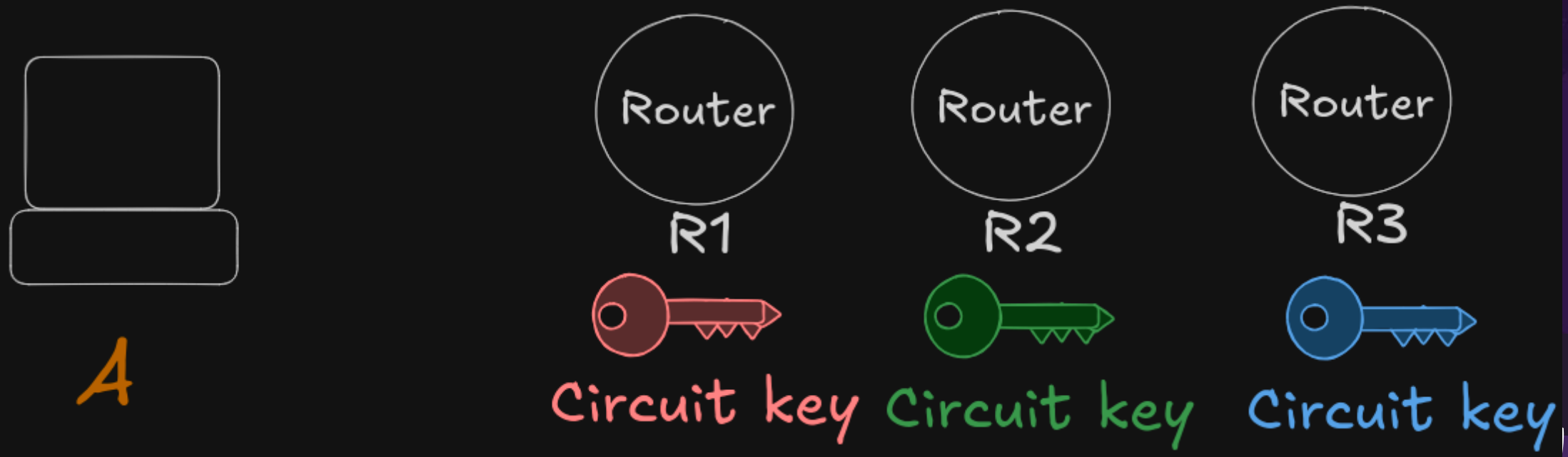

Router
R1
A
Onion key
Router
R1
R1
Message
A
Message
R1
A
Message
R1
R1
Message


Router
R1
A
Onion key
Router
R1
R1
Message
A
Message
R1
R1
Message
A
A
Message
R1
Circuit key

CIRCUIT KEY – IS A SHARED/SESSION KEY, IT MEANS ENCRYPTION END DECRYPTION WILL BE DONE WITH IT


Router
Router
R1
R2
A
Circuit key
Onion key
R2
Message
R1
Message
R2
A
Message
R1
A
Message
R1
Router
R1
R1
Message
R2
Router
R2
R2
Message

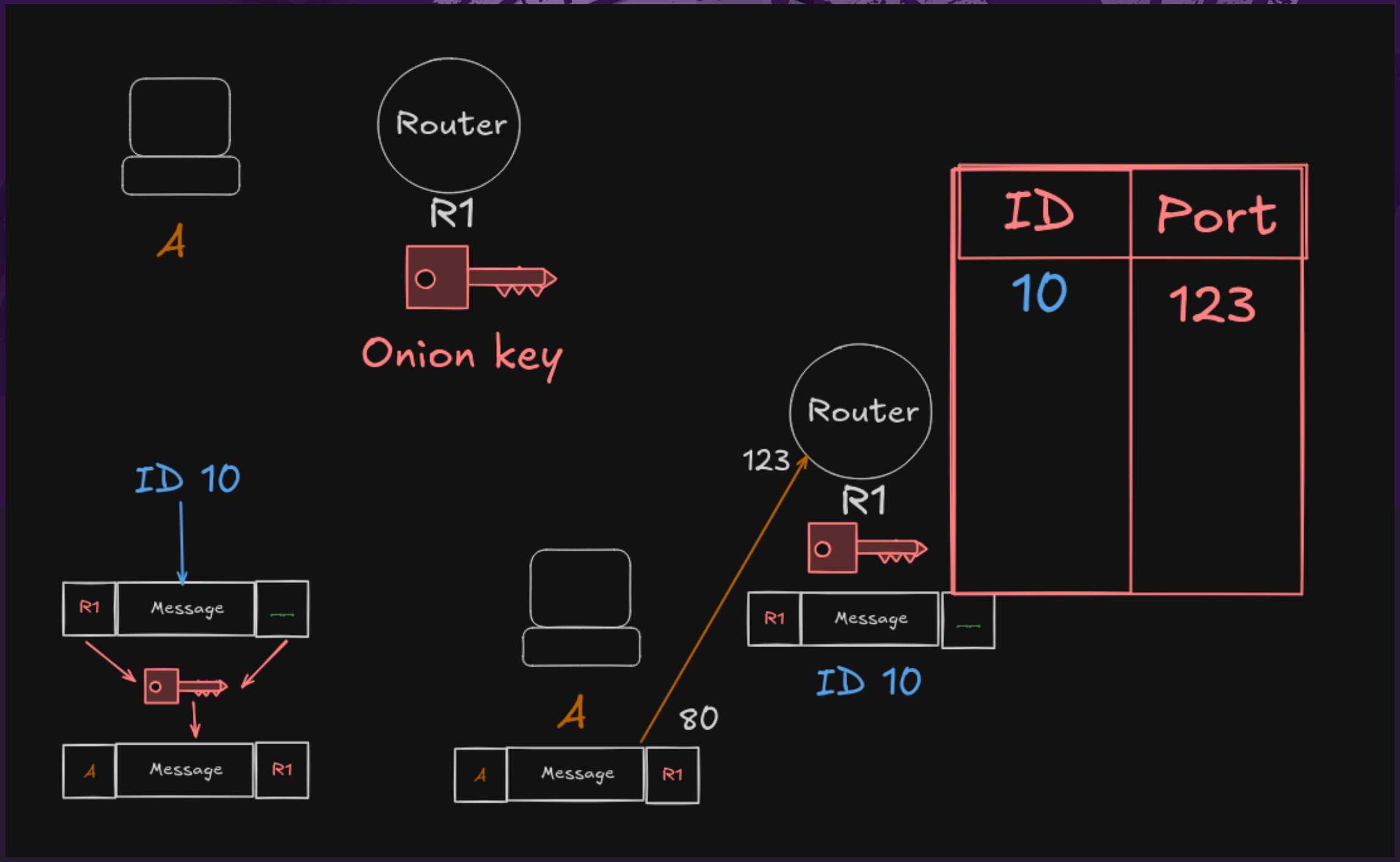

Router
R1
Router
R2
Router
R3
Circuit key
Circuit key
Onion key
A
R3
Message
R2
Message
R3
R1
Message
R2
A
Message
R1
A
Message
R1
Router
R1
R1
Message
R2
Router
R2
R2
Message
R3
Router
R3
R3
Message


Router
R1
Router
R2
Router
R3
A
Circuit key
Circuit key
Circuit key

# BUT HOW WILL PACKETS GO BACK? FROM SERVER TO CLIENT

WHEN WE WERE CREATING A CIRCUIT WITH EVERY PACKET, WE INCLUDED

CIRCUIT ID


A
Router
R1
Onion key
ID 10
R1
Message
A
Message
R1
A
Message
R1
80
123
Router
R1
R1
Message
ID 10
ID
Port
10
123

WHEN EXIT NODE GETS A MESSAGE FROM A SERVER IT SEES THE PORT AND KNOWS WHAT CIRCUIT AND KEY TO USE

separate table because you can use same circuit for different connections

| ID | Port |
|---|---|
| 10 | 123 |
| | |
| | |

| Port | Circuit key | Meta data |
|---|---|---|
| 123 | !2Fj3d!32Jkh612 | |
| | | |
| | | |

# THANK YOU FOR YOUR TIME

*Hope you liked it*