Anonymity and Tor

Outline

- Anonymity
- Proxies and VPNs
- Tor
  - Weaknesses: Timing attacks
  - Weaknesses: Collusion
  - Weaknesses: Distinguishable traffic
- Tor Onion Services
- Tor in Practice

Anonymity

Anonymity

- **Anonymity**: Concealing your identity
  - Anonymous communication on the Internet: The identity of the source and/or destination are concealed
- Anonymity is not confidentiality
  - Confidentiality hides the contents of the communication
  - Anonymity hides the identities of who is communicating with whom

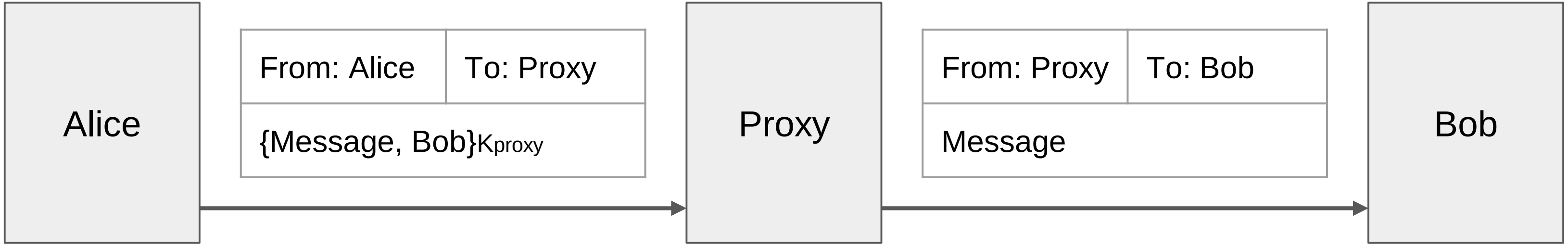
Anonymity on the Internet

- Anonymity on the Internet is hard
  - Difficult, if not impossible, to achieve on your own
  - Packets contain the source IP address and destination IP address
- Anonymity is easier for attackers
  - An attacker can hack into someone else's computer and send communications from that computer
  - We assume honest users won't hack into other computers
- Main strategy for anonymity: Ask someone else to send messages for you

Proxies and VPNs

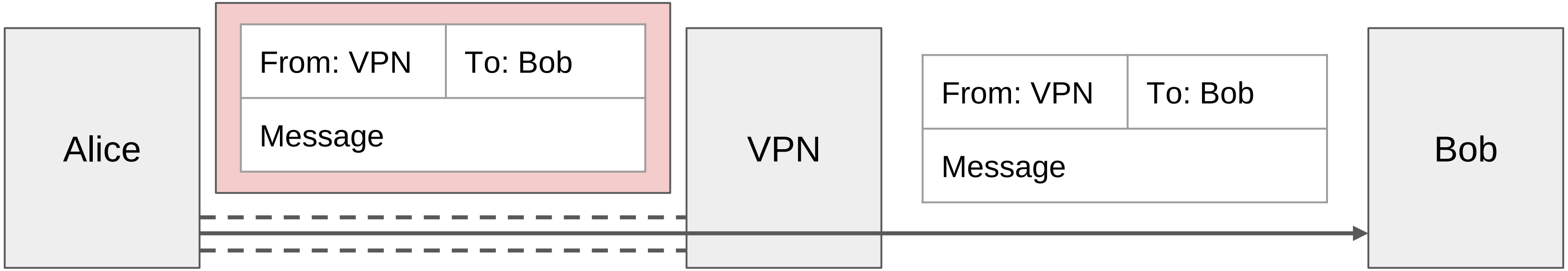
Proxies

- Alice wants to send a message to Bob
  - Bob shouldn't know the message is from Alice
  - An eavesdropper (Eve) cannot deduce that Alice is talking to Bob
- **Proxy**: A third party that relays our Internet traffic
  - Alice sends the message and the recipient (Bob) to the proxy, and the proxy forwards the message to Bob
    - The recipient's name (and optionally the message) is encrypted, so an eavesdropper does not see a packet with both Alice and Bob's identities in plaintext
  - Bob receives the message from the proxy, with no indication it came from Alice



Virtual Private Networks (VPNs)

- Recall VPNs: A virtual connection to an internal network
  - Allows access to an internal network through an encrypted tunnel
  - Creates an alternative use case: Appear as though you are coming from the virtually connected network instead of your real network!
    - Similar concept to proxies, but Alice directly sends packets as though coming from the VPN, wrapped in the VPN's layer of encryption
    - Proxies operate at the application layer, while VPNs operate at the network layer



Proxies and VPNs: Issues

- Performance
  - Sending a packet requires additional hops across the network
- Cost
  - VPNs can cost $80 to $200 per year
- Trusting the proxy
  - The proxy can see the sender and recipient's identities
  - Attackers might convince the proxy to tell them about your identity

Tor

Tor

- Idea: Send the packet through multiple proxies instead of just one proxy
- **Tor**: A network that uses multiple proxies (relays) to enable anonymous communications
  - Stands for **T**he **O**nion **R**outer
- Components of Tor
  - Tor network: A network of many **Tor relays** (proxies) for forwarding packets
  - Directory server: Lists all Tor relays and their public keys
  - Tor Browser: A web browser configured to connect to the Tor network (based on Firefox)
  - Tor onion services: Servers that can only be reached through the Tor network
  - Tor bridges: Tor relays that try to hide the fact that a user is connecting to the Tor network

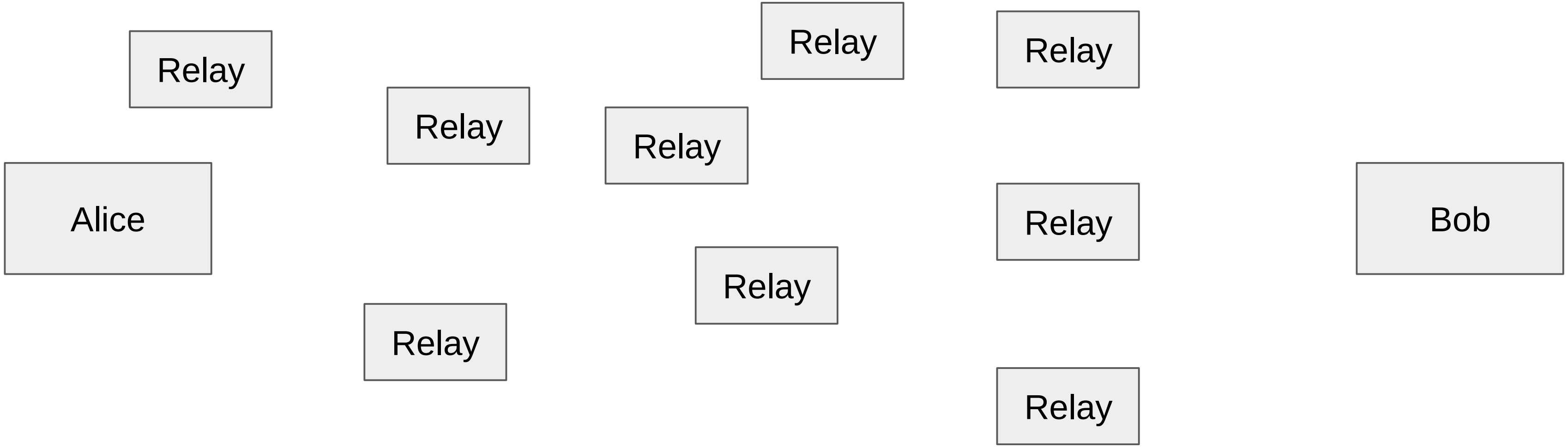
Tor Threat Model

- Security: Client anonymity and censorship resistance
  - Optional: Server anonymity with onion services
- Performance: Low latency (communication should be fast)
- Tor preserves anonymity against local adversaries
  - Example: An on-path attacker sees Alice send a message to a Tor relay, but not the final destination of the message
  - Example: The server should not know the identity of the client

Tor Circuits

- To communicate anonymously with a server, the Tor client forms a **circuit** consisting of 3 relays (by default)
    - Step 1: Query the directory server for a list of relays
    - Step 2: Choose 3 relays to form a Tor circuit
    - Step 3: Connect to the first relay, forming an end-to-end TLS connection
    - Step 4: Connect to the second relay *through* the first relay, forming an end-to-end TLS connection
    - Step 5: Connect to the third relay *through* the second relay, forming an end-to-end TLS connection
    - Step 6: Connect to the web server
        - If the web server is using HTTPS, then an end-to-end TLS connection will be formed through the third relay

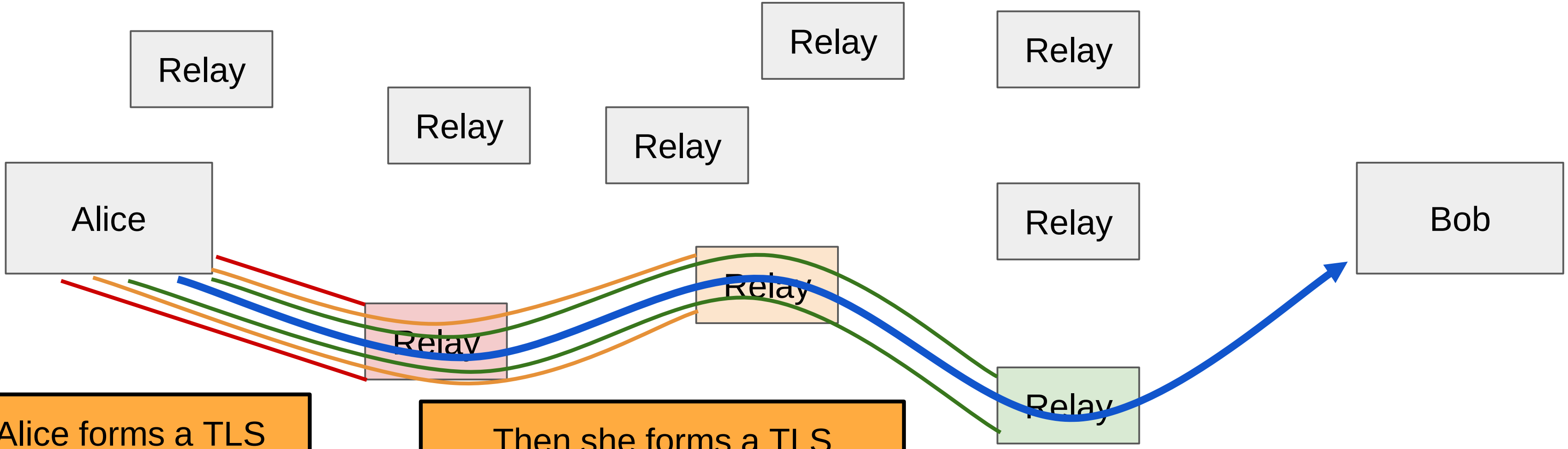
Tor Circuits



Suppose Alice wants to talk to Bob anonymously.

Alice queries the directory server and chooses 3 relays

Tor Circuits



Alice forms a TLS connection with the entry node

Then she forms a TLS connection with the second node, through the first node

Notice: Relay 1 is only relaying TLS packets. It doesn’t know the contents of the packets!

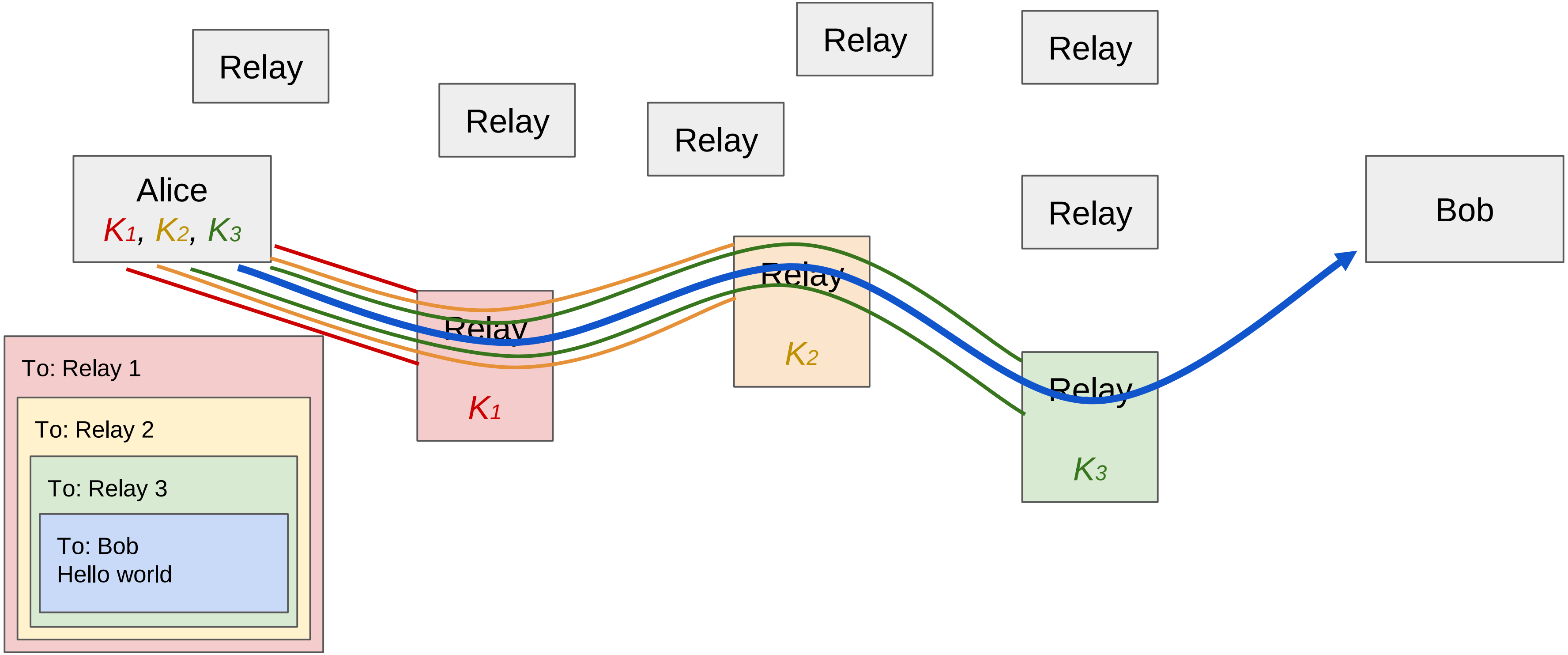
Then she forms a TLS connection with the exit node, through the second node

Finally, she connects to Bob (optionally forming a TLS connection with Bob)

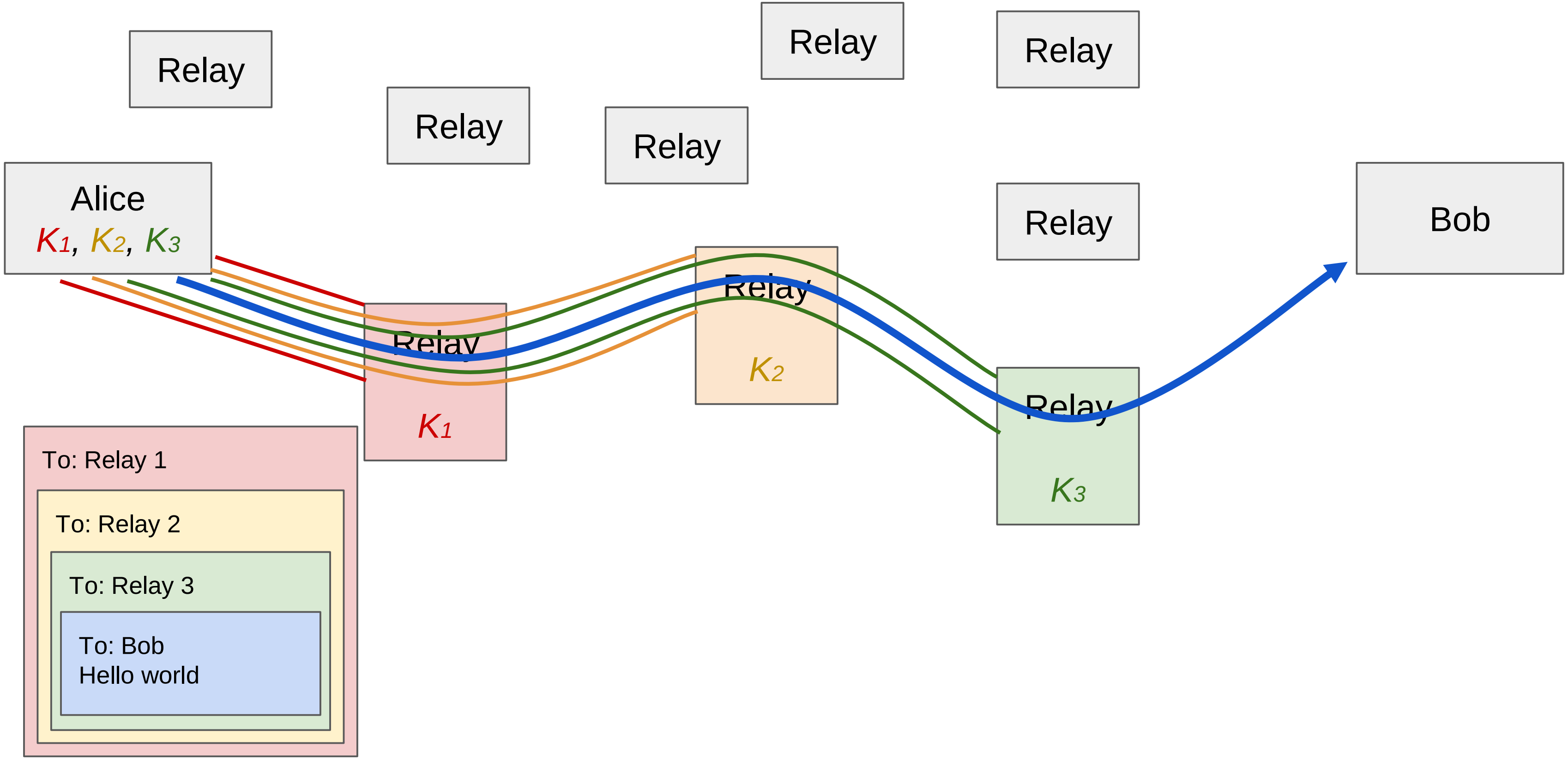
Tor Circuits

- Function of the relays:
  - Perform TLS handshakes when requested
  - When receiving a packet, decrypt using the key obtained through TLS
  - If the destination of the packet is another relay, forward the packet to the next relay
  - If the destination of the packet is an external server, forward the packet to that server

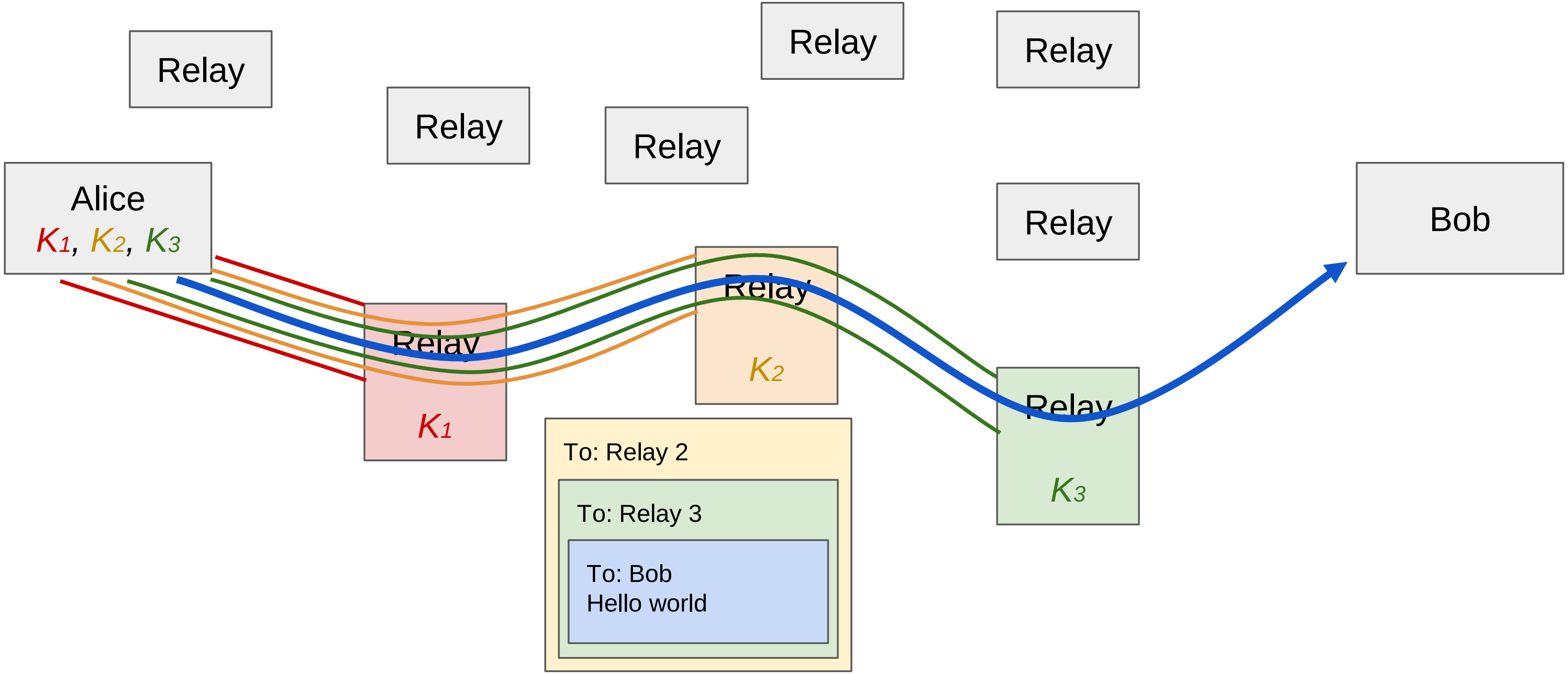
Tor Circuits



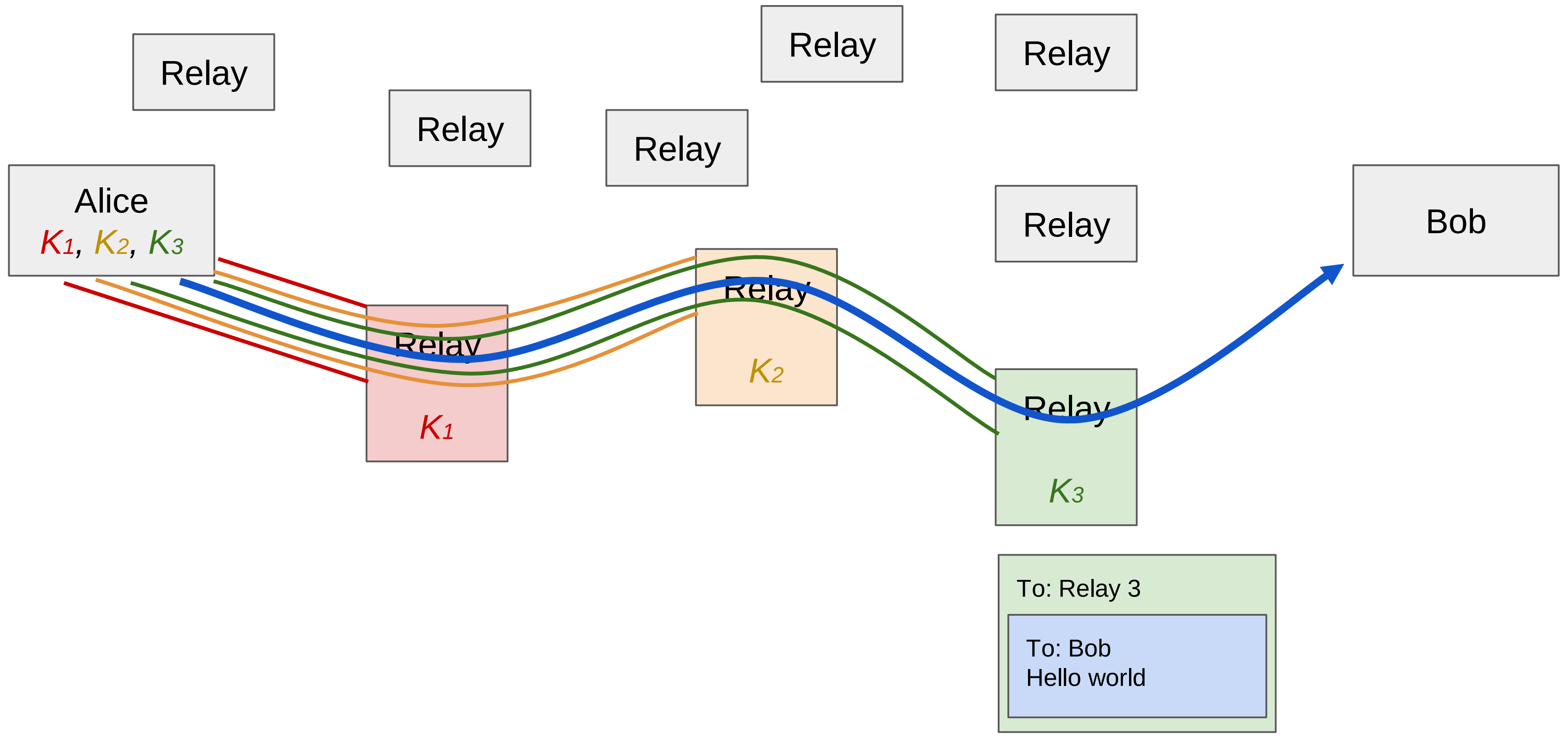
Tor Circuits



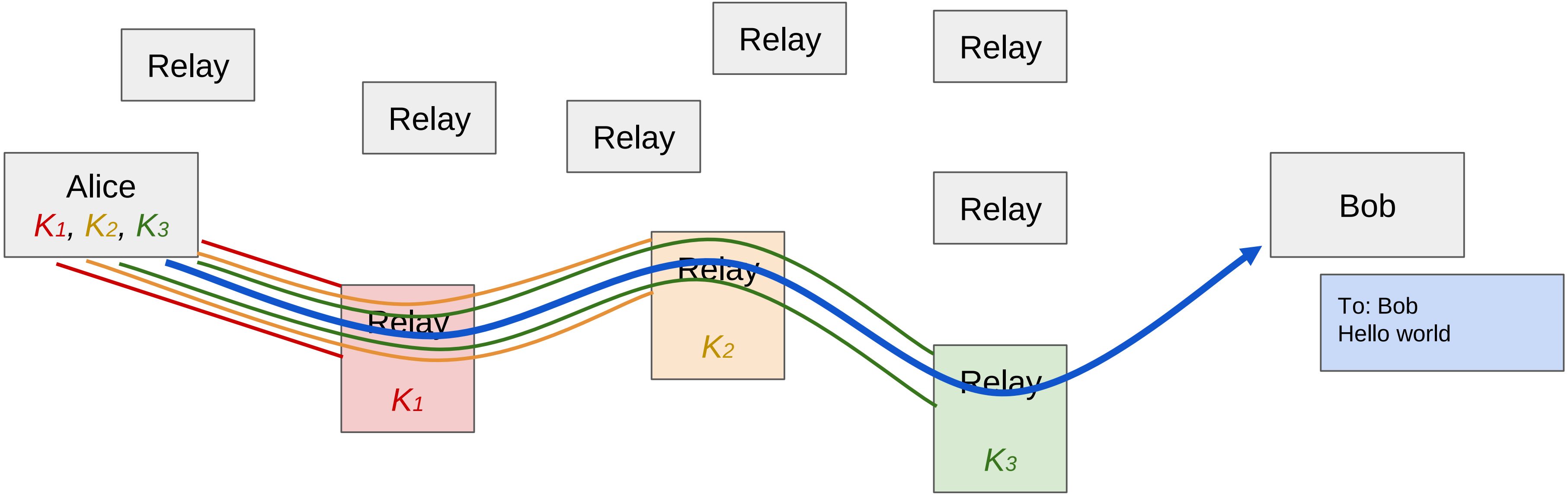
Tor Circuits



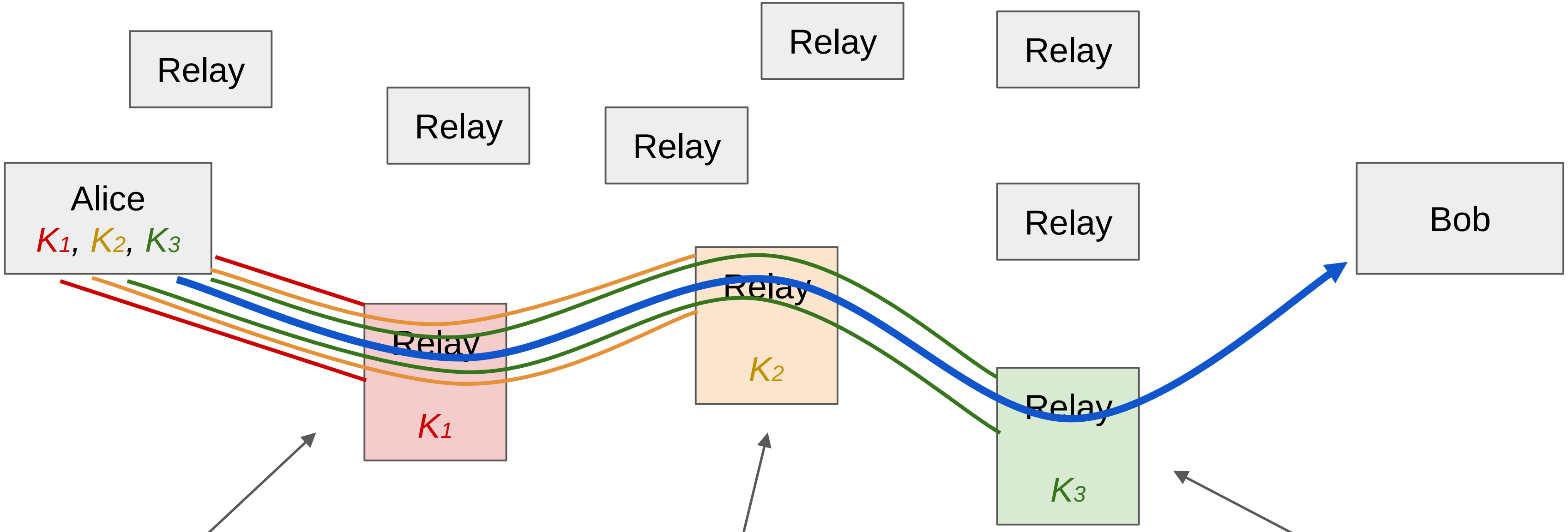
Tor Circuits



Tor Circuits



Tor Circuits



Relay 1 knows that Alice is using Tor but *not* who Alice is talking to

Relay 2 knows nothing (other than that someone is using Tor)

Relay 3 knows that someone is talking to Bob, but not who

Tor Exit Nodes

- Notice: The exit node can see the message and the recipient
  - Without collusion, the exit node doesn't know the sender
- The exit node is a man-in-the-middle attacker
  - If the user is not using TLS to connect to the end host (using HTTP), the exit node can see and modify the traffic
  - If the user is using TLS (using HTTPS), the exit node cannot see or tamper with the traffic

Tor Exit Nodes in Practice

- Administrators of Tor exit nodes often receive abuse complaints
    - Users complain to the exit node
    - Users complain to the Internet service provider (ISP), which complains to the exit node
- As a result, most Tor relays choose to only be entry or intermediate nodes, not exit nodes
    - Exit node bandwidth is the bottleneck in Tor, not internal bandwidth

Tor Weaknesses: Timing Attacks

- A network attacker who has a full (**global**) view of the network can learn that Alice and Bob are talking
  - Exploit a timing attack: Observe when Alice sends a message, when Bob receives a message, and link the two together
- Global adversaries are *outside* of Tor's threat model and are not defended against
  - Tor only defends against local adversaries with partial views of the network
  - Timing attacks could be defended against by delaying the timing of packets, but this violates Tor's performance goal

Tor Weaknesses: Collusion

- **Collusion**: Multiple nodes working together and sharing information
  - Collusion is adversarial (dishonest) behavior
  - Honest nodes should never share information with other proxies
  - If *all* nodes in the circuit collude, anonymity is broken
  - If *at least one* nodes in the circuit is honest, anonymity is preserved
- It is easy to form some amount of colluding nodes
  - An attacker can create hundreds nodes in the Tor network to increase the chance that your circuit consists entirely of the attacker's nodes!
- The more nodes we use, the more confident we are that they are not all colluding
  - It's much harder for 10 nodes to collude than for 2 nodes to collude
  - 3 nodes is generally considered good enough for industrial-grade security and is the default

Tor Weaknesses: Collusion

- Defense: **Guard nodes**
  - Guard nodes must have a high reputation and must have existed for a long time
  - Clients will always use a guard node as the entry node (by default) and the same guard node is used for a long period of time
    - Attackers’ nodes are unlikely to become guard nodes
    - Because clients use the same guard nodes for a long period of time, there is only a low chance that the client will switch to an attacker’s guard node

Tor Weaknesses: Distinguishable Traffic

- Tor does *not* hide the fact that you are using Tor
  - Example: A local adversary can see that you are sending packets to a Tor relay
  - Anonymity only works in a crowd
    - Example: A Harvard student sends an anonymous threatening message using Tor. The administrators notice that only one student on the Harvard network is using Tor!
    - Every Tor browser should be configured similarly, so network adversaries cannot distinguish any patterns in the packets
    - Tor browsers should resist tracking (e.g. no tracking cookies)

Tor Weaknesses: Distinguishable Traffic

- Defense: **Tor bridges**
  - Notice: Attackers can tell you are using Tor because they can see you are connecting to an entry node
    - Lists of entry nodes are publicly available
  - **Tor bridges** are entry nodes that are not available on any public list
    - Users request bridges from a separate directory, which will only give a few bridges to the user
    - There is no publicly available list of all bridges!
  - Censors can no longer block Tor based on IP addresses, but they can still distinguish traffic that looks like Tor traffic from normal traffic
- Defense: **Pluggable transports**
  - Pluggable transports change the appearance of the client’s traffic to the entry node (only for bridges)
  - Obfuscates the encrypted traffic to make it “look” more like normal web traffic

Tor Onion Services

Tor Onion Services

- Sometimes, the *server* wants to be anonymous, so no one knows where the server is located
- **Tor onion services**: Websites that are only accessible through the Tor network
  - Gives the server anonymity protection
  - Sometimes called the **dark web**
- Big idea: Route the server's traffic through the Tor network so that no one knows who the server is

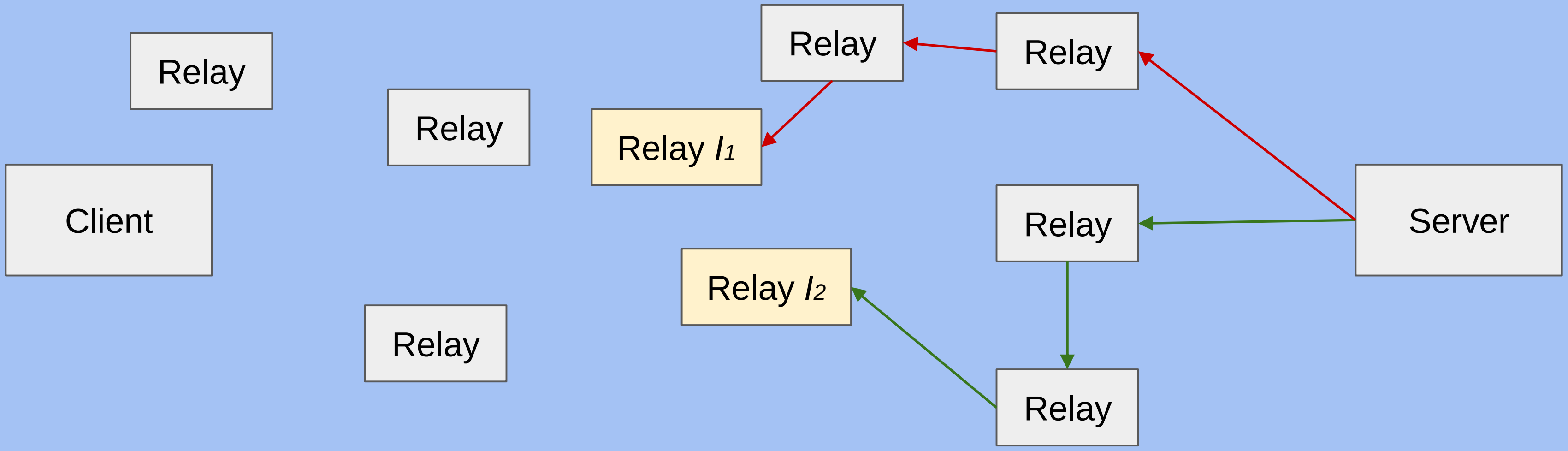
Tor Onion Services

- Recall: Standard domain names translate to IP addresses, which would break server anonymity
    - Instead, identify servers using the hash of the service's public key encoded as a .onion address
    - Example: **`http://pwoah7foa6au2pul.onion`**
    - Example: **`https://facebookcorewwwi.onion`** (Facebook brute-forced key pairs until they found one with a human-readable hash)

Tor Onion Services

- Connecting to onion services is a little more involved, since you can't just contact the server after the final hop
- First, the server needs to publish how to contact the server
  - Step 1: The server chooses a set of nodes to be **introduction points** and forms a Tor circuit to each of them
  - Step 2: The server publishes its public key and its introduction points to a publicly available directory
    - This directory is set up such that no one knows the full list of services (or .onion addresses)
    - Because of this, you must have come across the .onion address somehow (a friend sent it to you, someone compiled a list of addresses, etc.)

Tor Onion Services



The server chooses nodes to be the introduction points

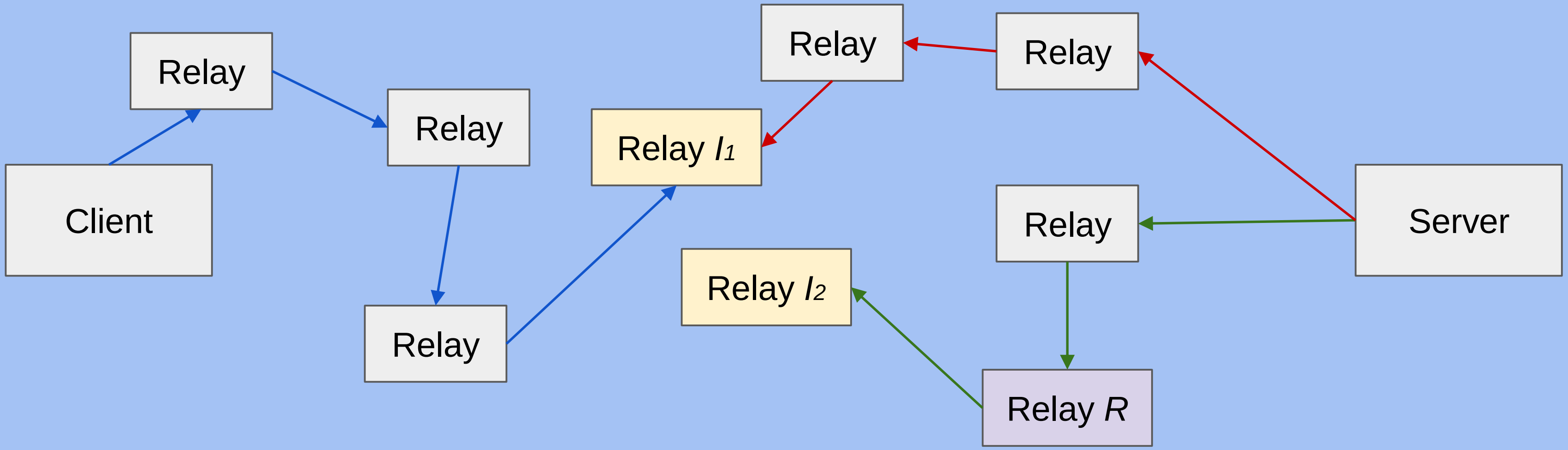
It publishes its key and the introduction points

Hash	**Public Key**	**Introduction Points**
…	…	…
expuy….onion	*PK*	I_1, I_2

Tor Onion Services

- Now, the client connects to the server
  - Step 1: The client queries the directory using the hash of the public key to get the server's full key (not just its hash) and the introduction points
  - Step 2: The client chooses an introduction point and forms a Tor circuit to it
  - Step 3: The client chooses a **rendezvous point** and **secret** used to communicate to the server, encrypts them with the server's public key, and sends them to the introduction point, which relays them to the server
  - Step 4: The client and server both form Tor circuits to the rendezvous point and perform an end-to-end TLS handshake, and the server sends the decrypted secret to the client to authenticate itself

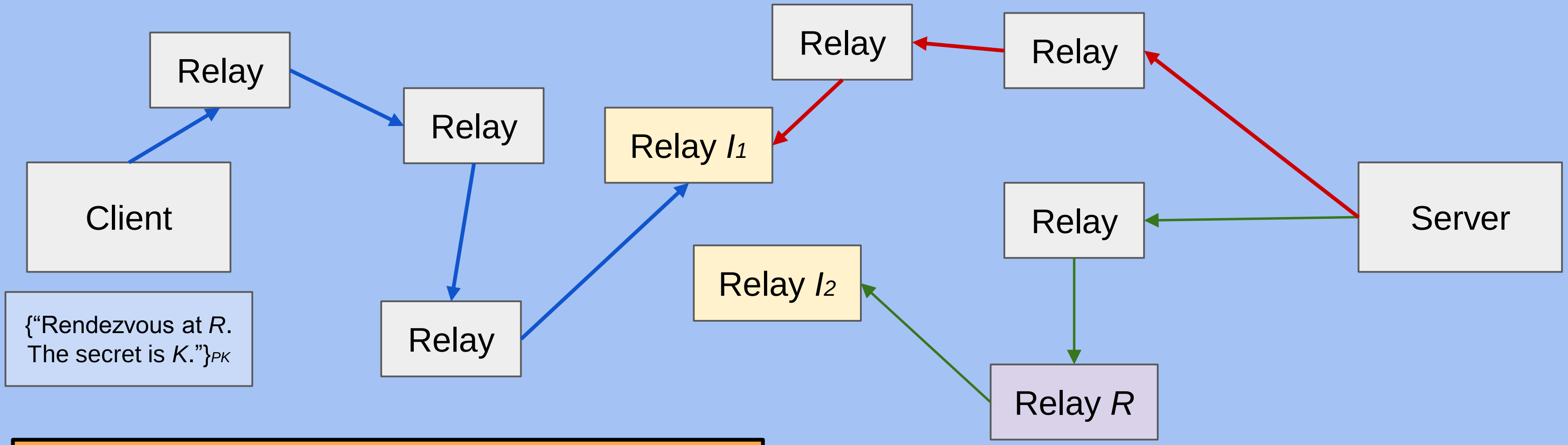
Tor Onion Services



The client queries the directory and connects to an introduction point

Hash	Public Key	Introduction Points
…	…	…
expuy….onion	*PK*	I_1, I_2

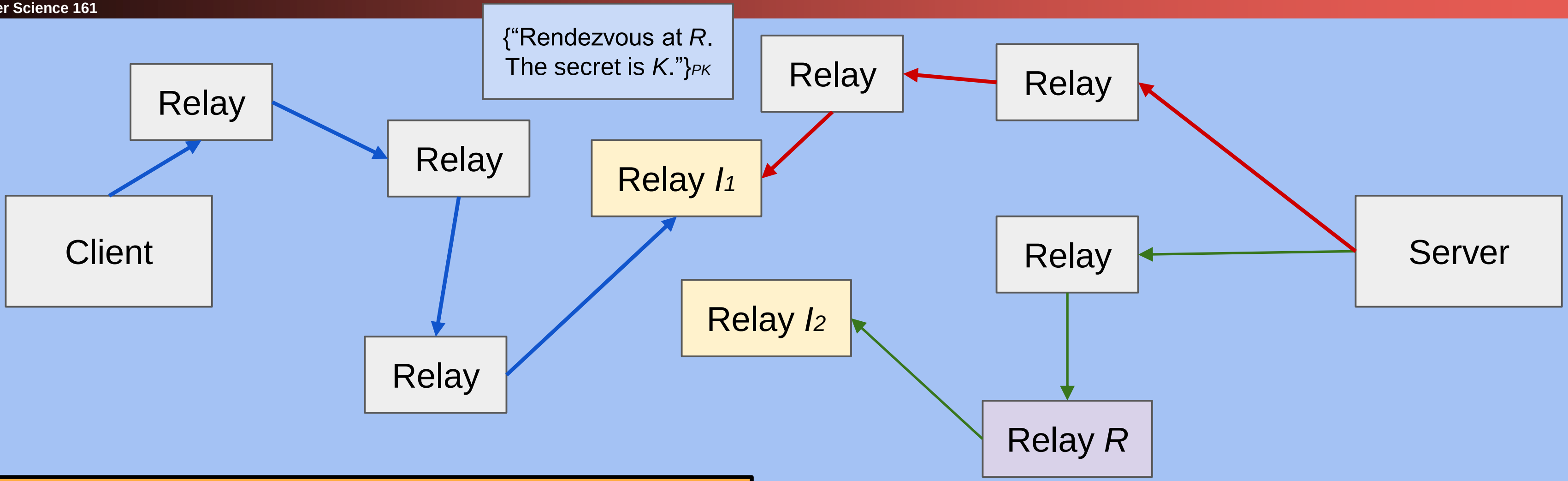
Tor Onion Services



The client chooses a rendezvous point and secret, encrypts using the public key, and sends them to the server through the introduction point

Hash	**Public Key**	**Introduction Points**
...	...	...
expuy….onion	*PK*	I_1, I_2

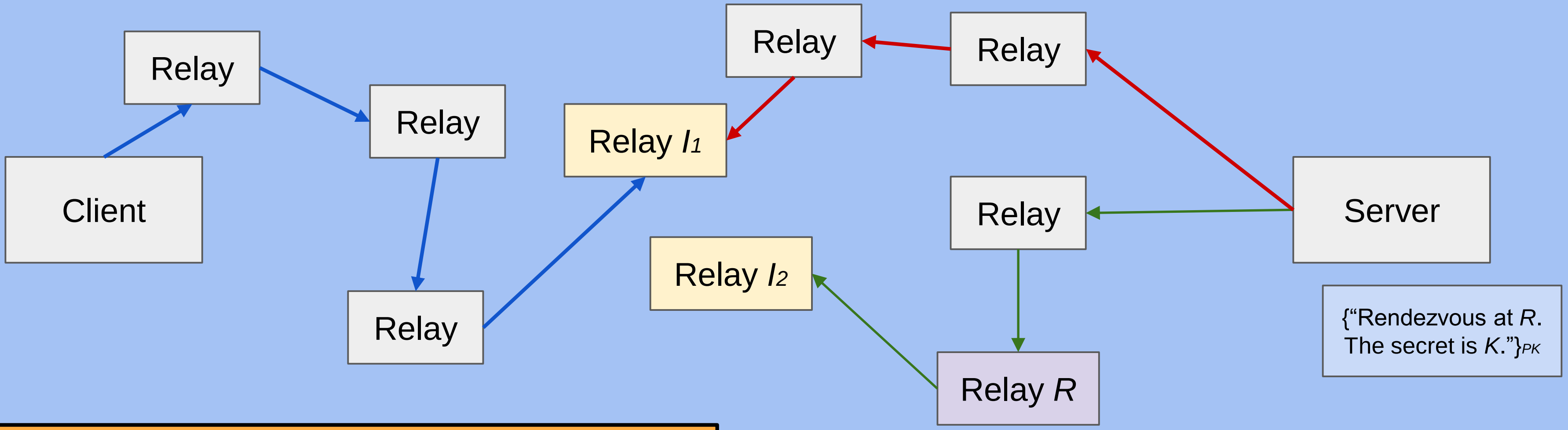
Tor Onion Services



The client chooses a rendezvous point and secret, encrypts using the public key, and sends them to the server through the introduction point

Hash	**Public Key**	**Introduction Points**
...	...	...
expuy….onion	*PK*	I_1, I_2

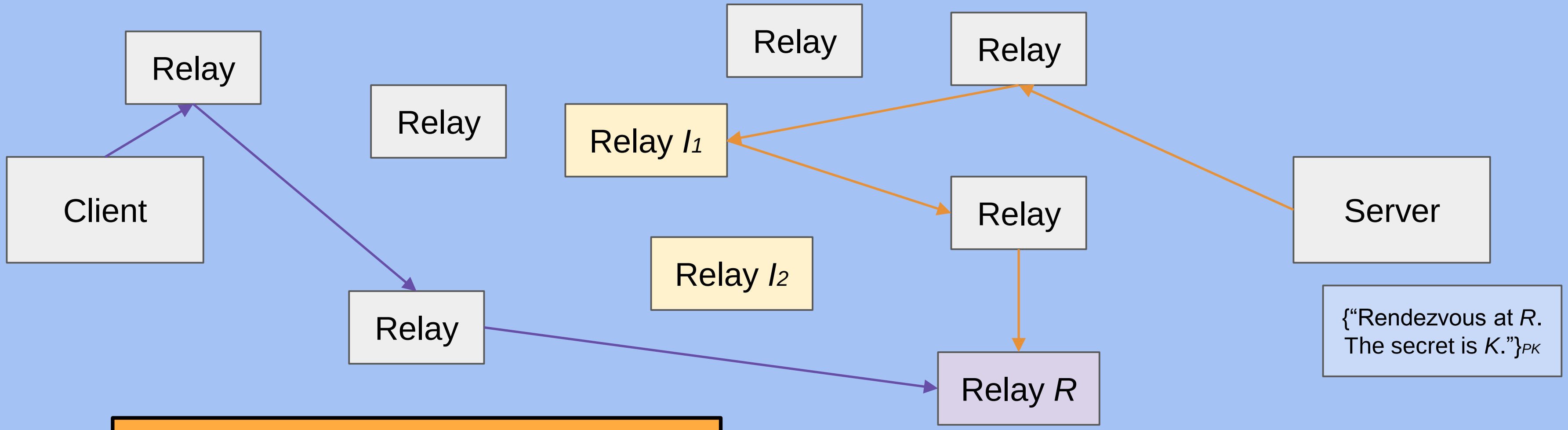
Tor Onion Services



The client chooses a rendezvous point and secret, encrypts using the public key, and sends them to the server through the introduction point

Hash	**Public Key**	**Introduction Points**
...	...	...
expuy….onion	*PK*	I_1, I_2

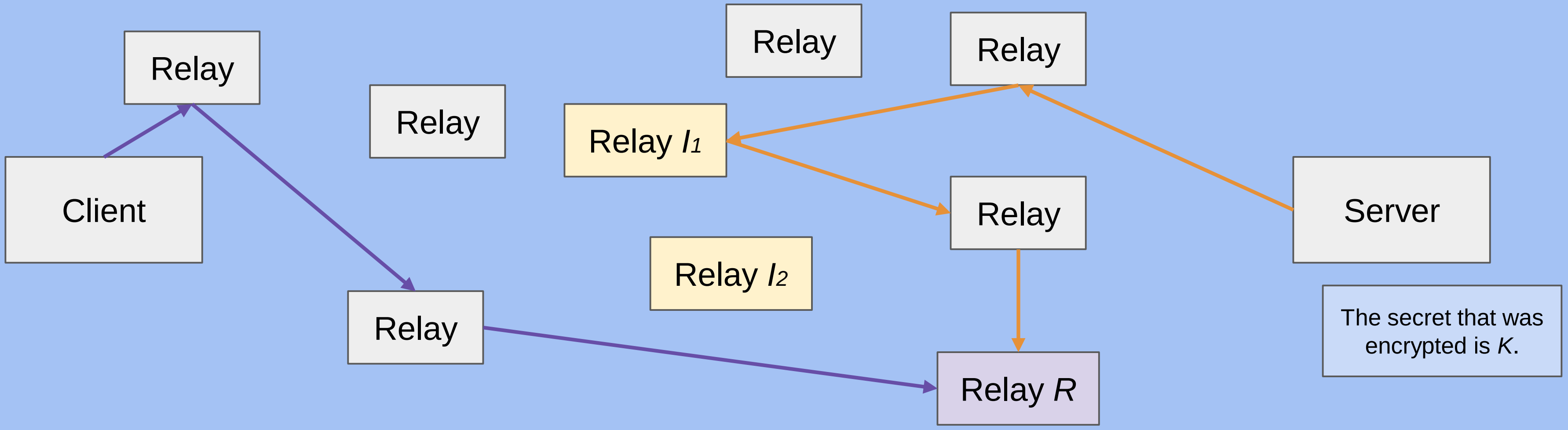
Tor Onion Services



The client connects to the rendezvous point and waits

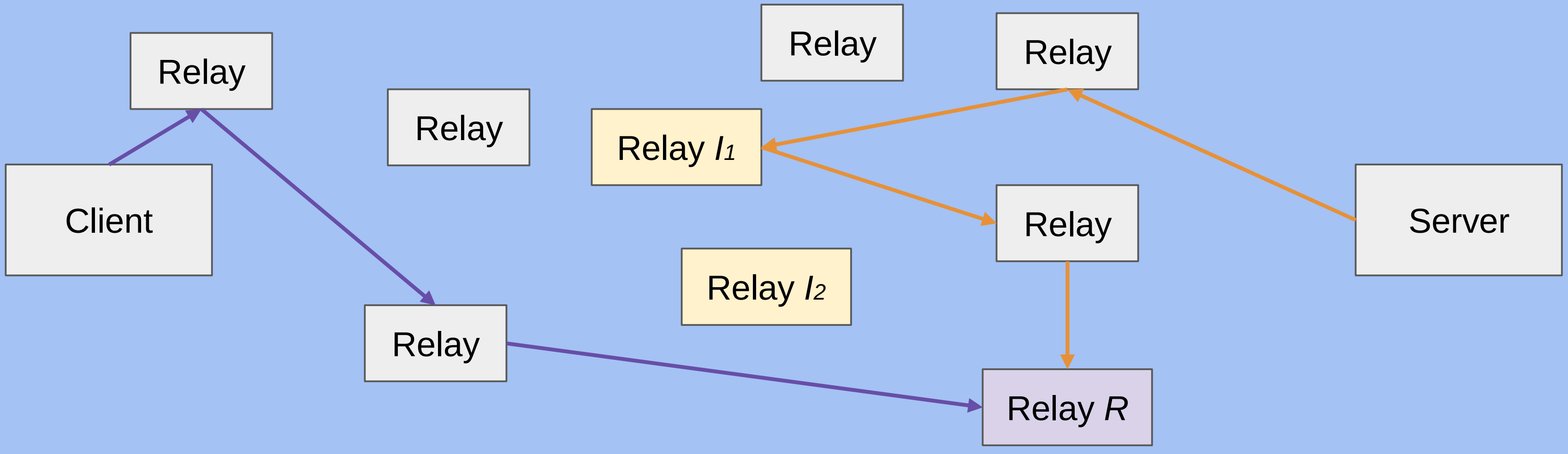
The server connects to the rendezvous point

Tor Onion Services



The client and server perform a handshake, and the server sends the decrypted secret

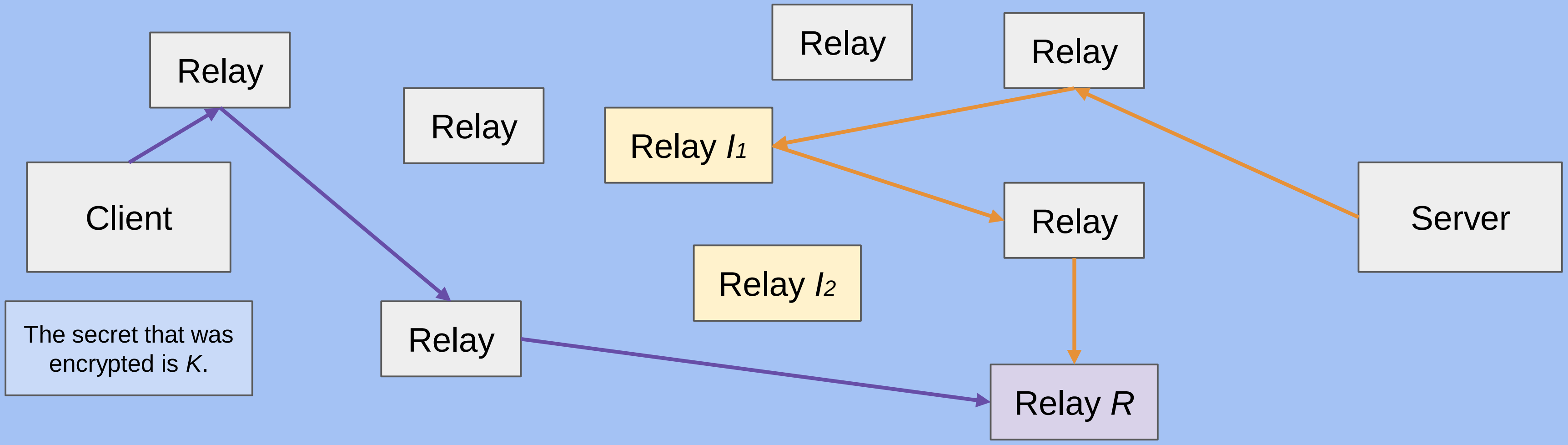
Tor Onion Services



The secret that was encrypted is *K*.

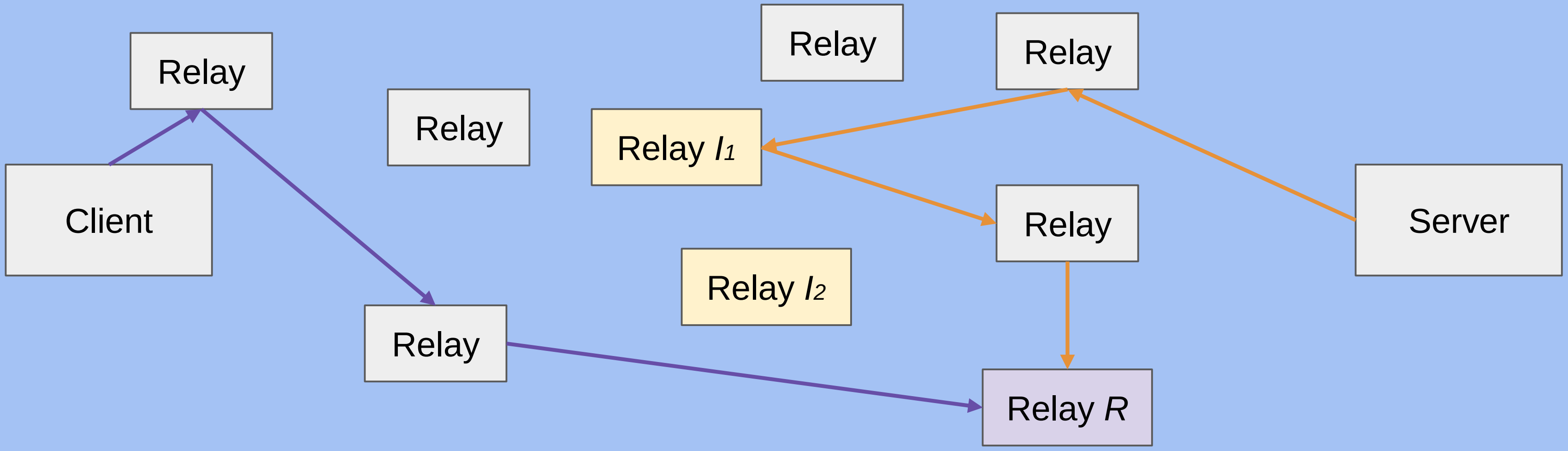
The client and server perform a handshake, and the server sends the decrypted secret

Tor Onion Services



The client and server perform a handshake, and the server sends the decrypted secret

Tor Onion Services

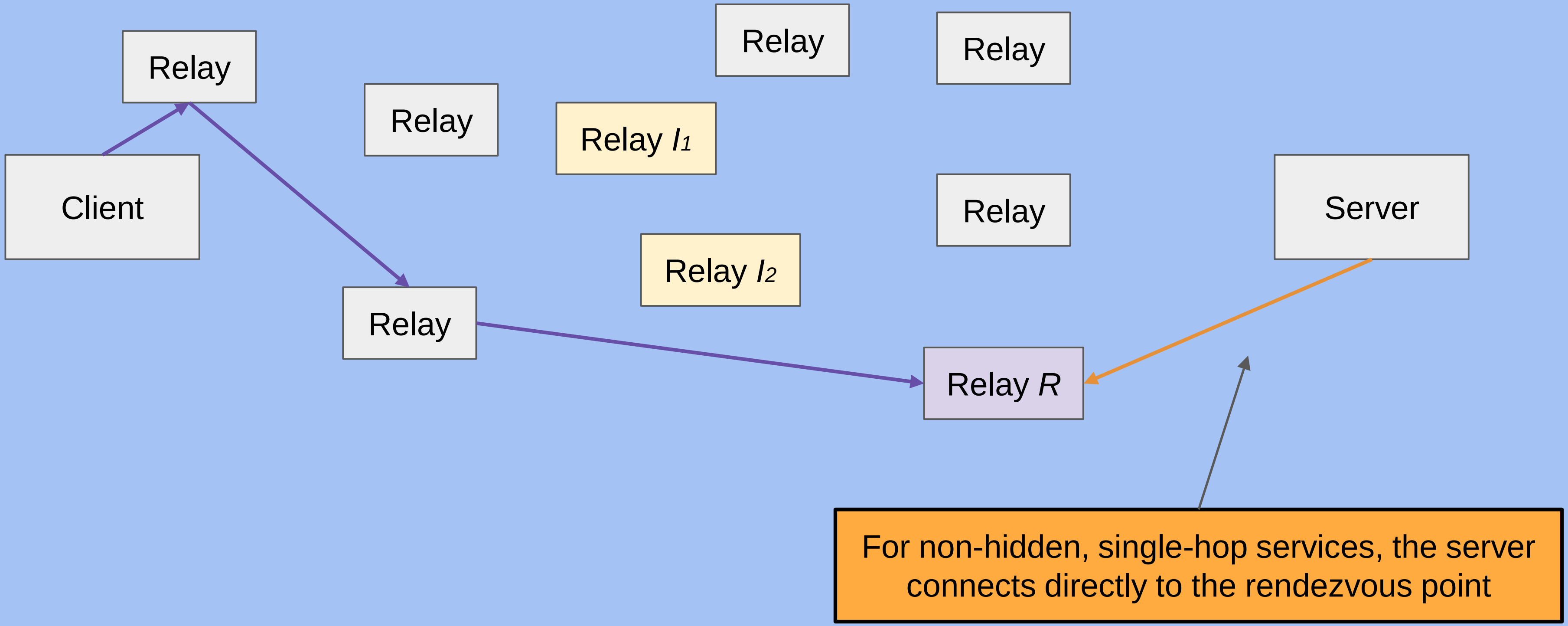


Notice: The client and server never directly communicate, and the introduction and rendezvous points don’t know the client or the server

Tor Onion Services

- Truly hidden onion services
  - Provides Tor’s anonymity guarantees for both the client and the server, instead of just the client
  - Performance impact: Traffic travels through 6 hops in Tor network!
- Non-hidden onion services
  - Servers can opt to skip its side of the Tor circuits
    - No more anonymity for the server!
  - Better performance: Same performance as a public service
  - Better performance: Not limited by exit node bandwidth
  - Better security: No longer rely on exit nodes being honest
  - Useful for public services with an onion alternative (e.g. Facebook, DuckDuckGo, etc.)

Tor Single-Hop Onion Services



Tor in Practice

Tor Tradeoffs

- Benefit: Free to use
  - Tor is mostly funded by the US government
  - Users “pay” by providing traffic for other users to hide in (recall: you don’t want to be the only user on the network using Tor)
- Drawback: Exit nodes are a man-in-the-middle attacker
  - However, the regular Internet is full of MITMs, as well (e.g. your ISP)
- Drawback: Performance
  - Latency is significantly worse: Packets need to make more hops across the network
- Drawback: Full anonymity requires usability tradeoffs
  - All Tor browsers need the exact same configuration, so they don’t save your history
  - They even recommend keeping the browser window size constant, which can be annoying!

Tor for Censorship Resistance

- Because Tor hides the sites a user is connecting to, it is useful for bypassing censorship
  - Functions similarly to bypassing censorship using a VPN or proxy
- Censors can easily block access to all public Tor entry points
  - Bridge services provide a set of entry points that aren't listed publicly anywhere, so they can't be blocked by IP
- Censors can block traffic that looks like Tor traffic
  - Pluggable transports make traffic look more like normal web traffic
- Censors can pretend to be a Tor client to see if an endpoint is a Tor node
  - More recent pluggable transports distribute a shared secret, not known to active probers
  - Some pluggable transports deliberately rely on cloud services, so censors have to block important web services (like Google Cloud Platform, Amazon Web Services, etc.) to block Tor
- Arms race between Tor and censors

Hosting Illegal Services on Tor

- Tor onion services are often used for services widely considered illegal around the world
  - Legitimate hosting services like Cloudflare will refuse to host these services
  - Most countries will take legal action against these services if hosted on the regular web
- **Dark markets**: Marketplaces for buying and selling illegal goods
  - Transactions processed with a censorship-resistant currency like Bitcoin
    - Services like PayPal will refuse to process illegal transactions
  - Ratings system with mandatory feedback
  - Escrow service to handle disputes between sellers and buyers
  - Can only be accessed as a Tor onion service
- **Cybercrime forums**: Websites for discussing illegal activity

History of Dark Markets

- The first dark market: Silk Road
  - Founded in 2011 as a libertarian marketplace (no regulations)
  - Used for buying and selling illegal drugs
  - Taken down in October 2013
  - Its founder was arrested
- Modern dark markets follow the Silk Road template
  - Most common product: drugs
    - Mostly marijuana, MDMA, and stimulants
    - Some opioids and psychedelics
  - Most revenue is comes from a few major sellers and a few major markets
    - If a seller or market is taken down, another one takes its place

Modern Dark Markets

- Hard to find information about where dark markets are located
    - Legitimate websites (e.g. Reddit) will remove dark market links
    - Legitimate websites with information about dark markets (e.g. DeepDotWeb) get taken down
    - Information about dark markets is usually available through Tor onion services (e.g. Dread, a Reddit clone)
- Dark markets usually include sales volume information from the mandatory reviews
    - Security researchers crawl dark markets for prices and sales volumes to estimate the size of dark markets
    - Modern dark markets size: between USD$300,000 and USD$500,000 per day in sales
    - Latest peak: Close to USD$1,000,000 per day
    - Market size has been relatively steady for years, and is not growing

Dark Market Scams

- The reputation system tries to defend against scams
  - Someone selling misleading or fake products would have low ratings
- Exit scam: Sacrificing reputation for short-term profit
  - Spend some time building up a positive reputation with legitimate sales
  - Hold a big sale, forcing buyers to finalize their transactions early
  - Find a way to bypass escrow (because of “problems”)
  - Take the money and run
- Entire markets can be scams
  - Example: “Sheep marketplace”

Summary: Tor

- Anonymity conceals an individual's identity, but this can be difficult to achieve on the web
- Proxies and VPNs relay traffic through a single machine to conceal your identity from the end server
  - Issue: The single relay knows who you are and what you are doing, which is not anonymous!
- Tor routes your traffic through multiple machines
  - No one machine knows both who you are and what you are doing
  - Circuits are established by performing TLS handshakes with three nodes, nesting encrypted channels
  - Exit nodes can be a MITM since they are the final relay before traffic is sent to the server
  - Weakness: Timing attacks allow global adversaries to see when packets exit and leave the Tor network, deanonymizing users
  - Weakness: Collusion between nodes can deanonymize users by working together
    - Defense: Guard nodes
  - Weakness: Tor traffic is distinguishable from normal traffic, allowing it to be censored and blocked
    - Defense: Bridges and pluggable transports

Summary: Tor

- Onion services provide anonymity for the server, in addition to the client
  - Routes the server's traffic through the Tor network to anonymize the server
- Tor in practice
  - Provides anonymity in exchange for additional potential for MITM attacks (when not using HTTPS), performance, and usability
  - Often used to evade censorship
    - Tor and censors are in an arms race
  - Illegal services often use Tor because it conceals their identity from authorities