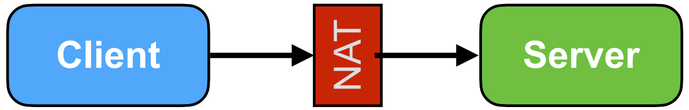
NAT Traversal and Peer-to-Peer Connection Establishment

- NAT Traversal Concepts
  - Binding discovery
  - Candidate exchange
  - Probe for connectivity
- Costs and Limitations

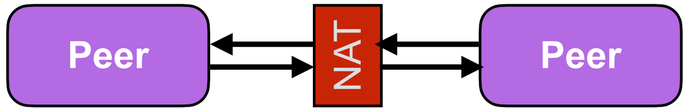
Peer-to-peer NAT Traversal Concepts

- NATs support outbound connections from client to server

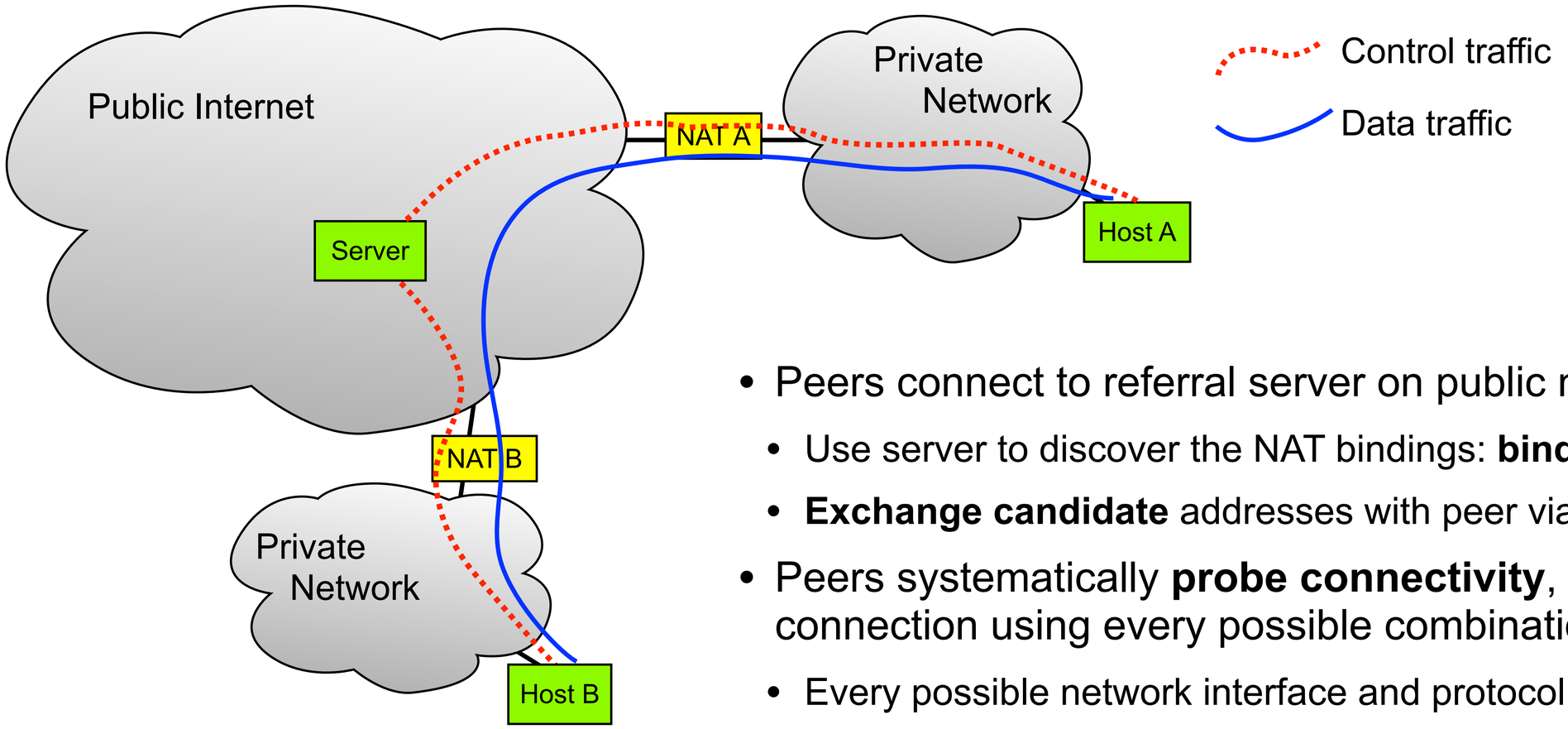


- Incoming connections fail, since NAT cannot know how to translate the incoming packets

- Peer-to-peer connections can succeed if both NATs think a client server connection is being opened, and the response is coming from the server

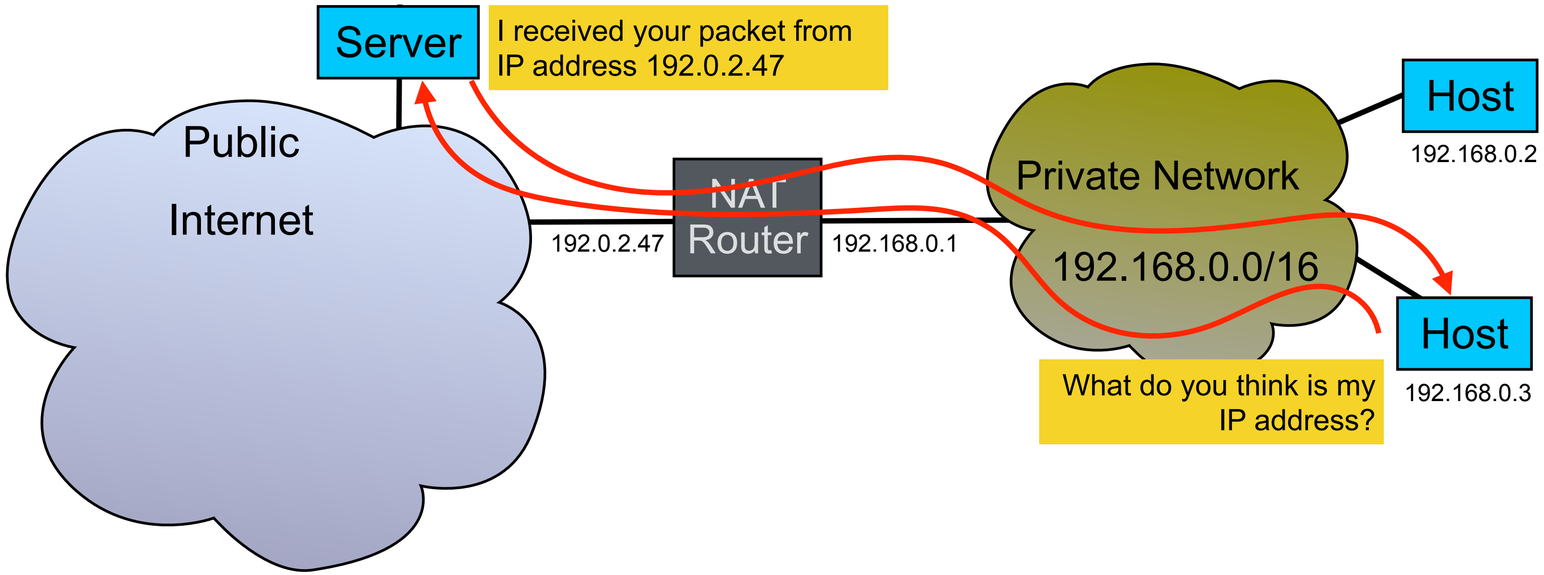


Peer-to-peer NAT Traversal Concepts



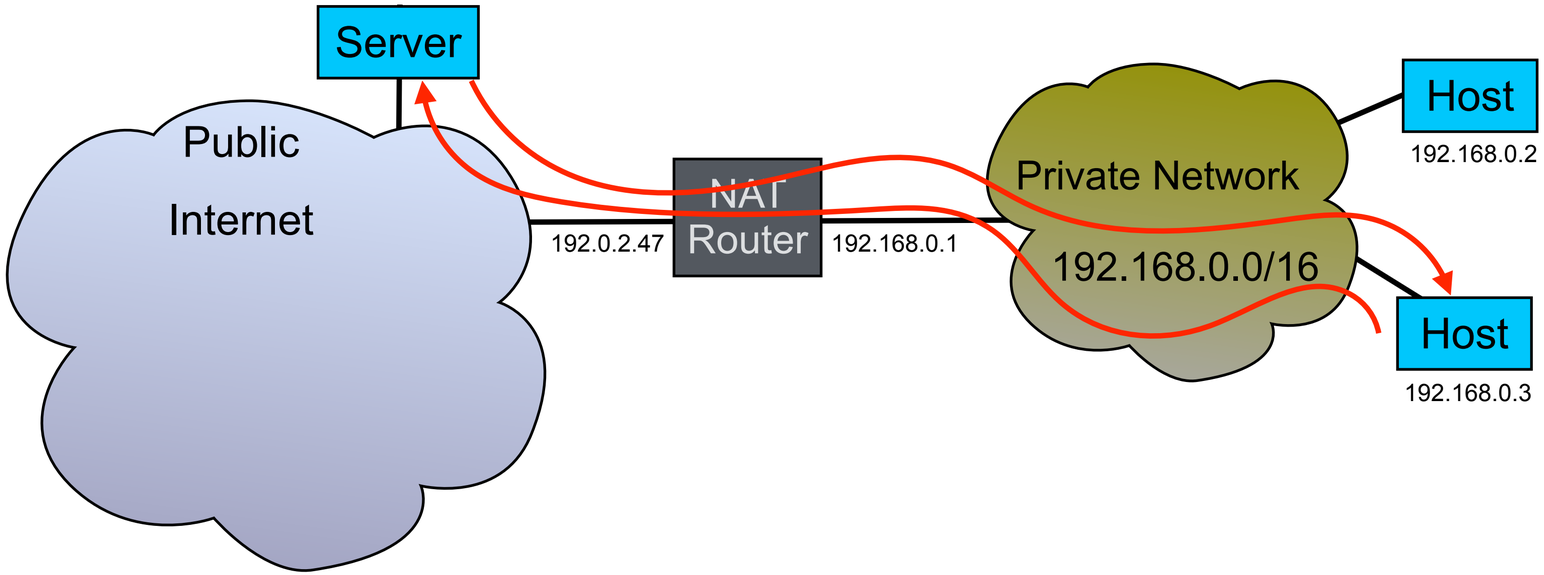
- Peers connect to referral server on public network
  - Use server to discover the NAT bindings: **binding discovery**
  - **Exchange candidate** addresses with peer via the referral server
- Peers systematically **probe connectivity**, try to establish a connection using every possible combination of addresses
  - Every possible network interface and protocol, mapped and local
  - Complex and generates significant traffic overhead

Binding Discovery (1/2)



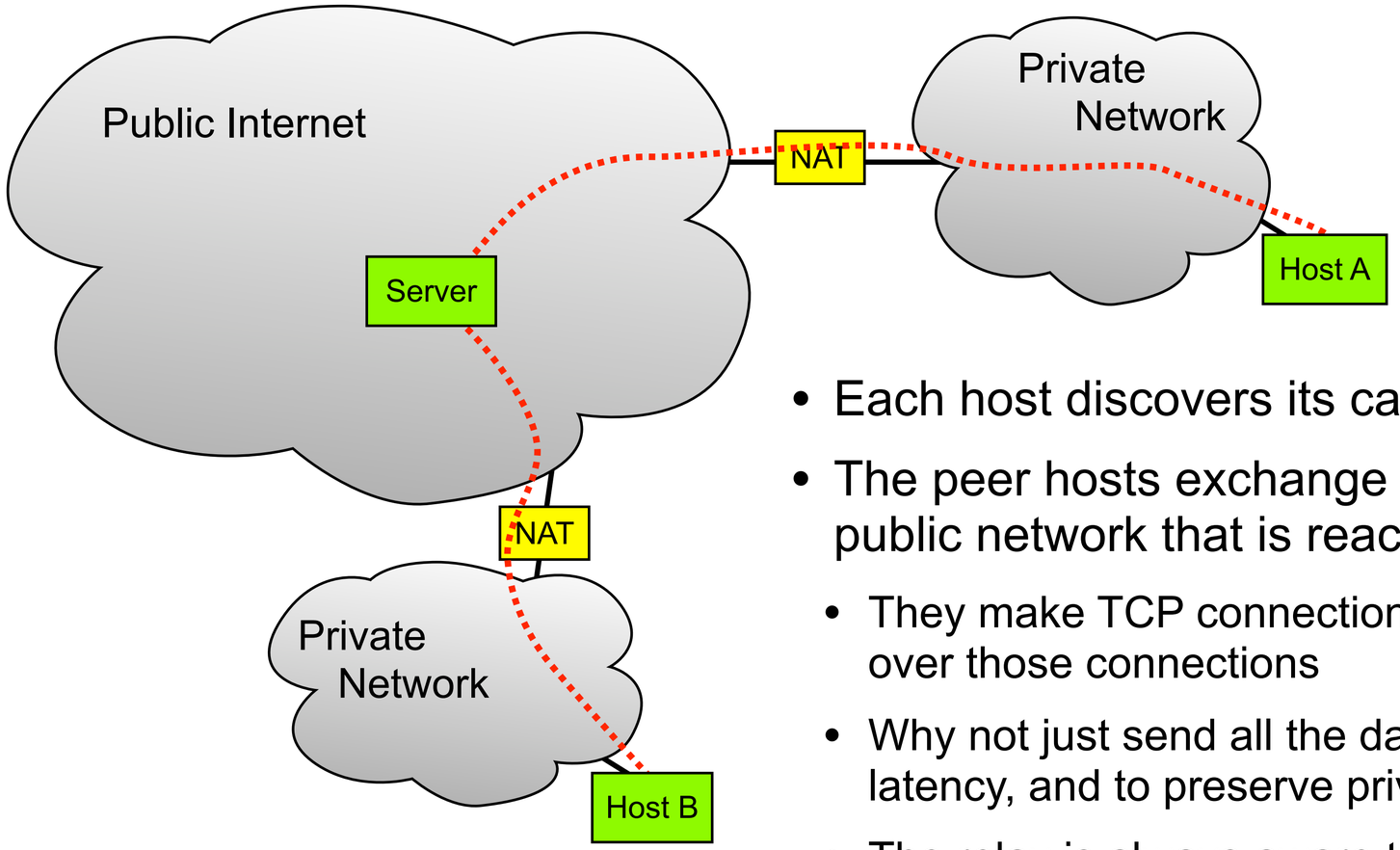
- Packets sent from a host on a private network to a server on the public network will have source IP address and port translated
- The server can see the translated address/port, and send a reply back to the host telling it the address its packets appear to come from – the server reflexive address
- Known as **NAT binding discovery**
- The **STUN Protocol** (Session Traversal Utilities for NAT) is a standard binding discovery mechanism https://datatracker.ietf.org/doc/rfc8489/

Binding Discovery (2/2)



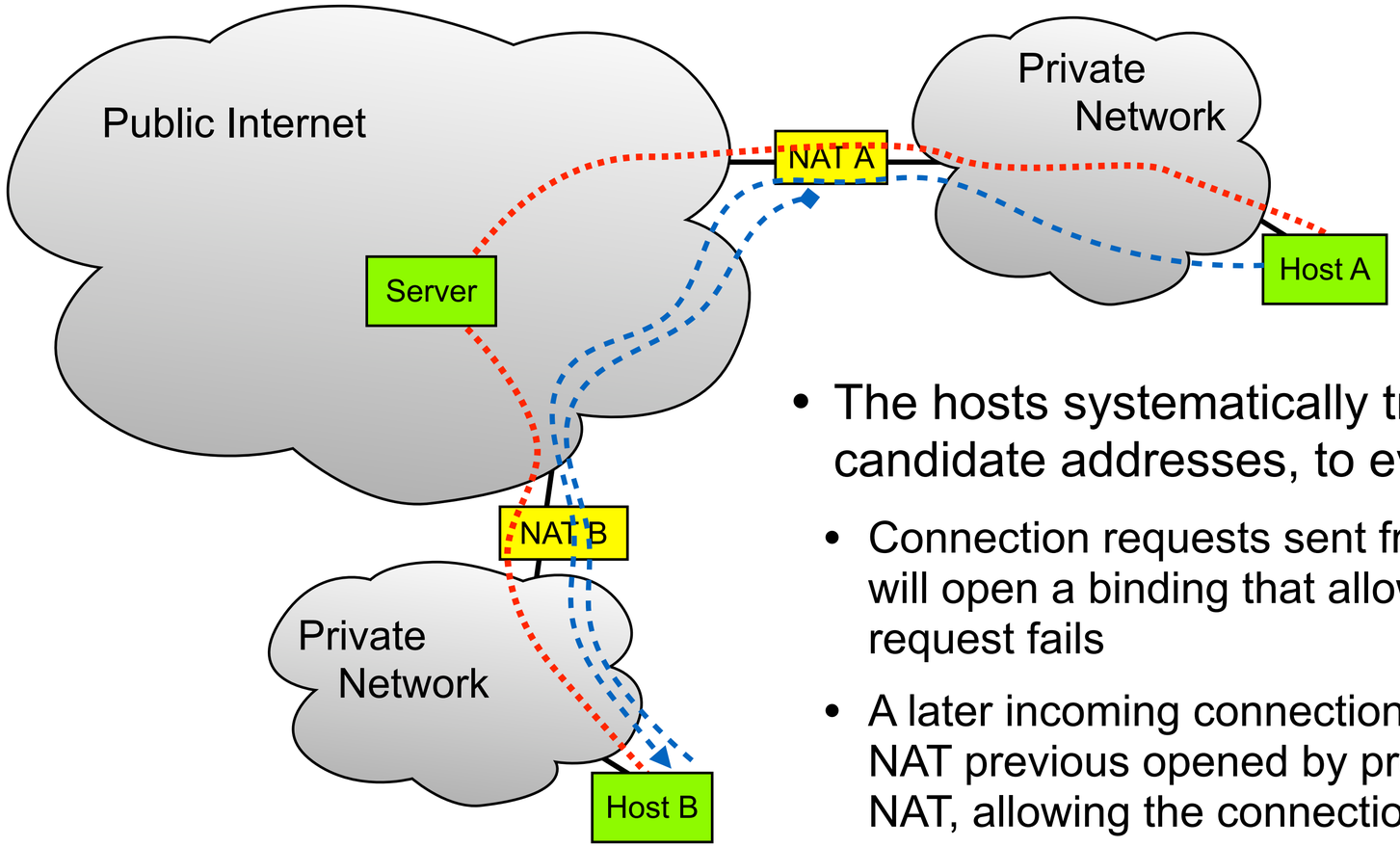
- Host attempts to find every possible **candidate** IP address, on which it might be reachable
  - The IPv4 and IPv6 addresses of each of its interfaces (Wi-Fi, Ethernet, 4G, …)
  - For each address, any server reflexive addresses discovered using STUN from that address
  - Any relayed addresses (e.g., via a TURN or SOCKS proxy, VPN server, etc.)

Candidate Exchange



- Each host discovers its candidate IP addresses/ports
- The peer hosts exchange candidates, using server on the public network that is reachable by both as a relay
  - They make TCP connections to the relay server and exchange data over those connections
  - Why not just send all the data via the relay server? To reduce latency, and to preserve privacy
  - The relay is always aware that communication is being attempted; this *communication metadata* is potentially sensitive information

Probe for connectivity: The ICE Algorithm



- The hosts systematically try **`connect()`** from each of their candidate addresses, to every candidate address of their peer
  - Connection requests sent from a host that passes through a NAT will open a binding that allows a response, even if the connection request fails
  - A later incoming connection request that reaches the port on the NAT previous opened by previous outgoing request will pass the NAT, allowing the connection to succeed
  - The hosts then exchange data over the path to confirm success
- **The ICE algorithm** describes how to do this [RFC 8445]

Peer-to-peer NAT Traversal

- Binding discovery and systematic connection probing is complex, slow, and generates a lot of unnecessary traffic
- Effective for UDP traffic
  - Developed to support VoIP applications, that send UDP packets
  - Connectionless nature of UDP means NATs tend to be permissive about allowing incoming packets, provided they reach the correct port
- Less effective for TCP connections
  - NATs often require an exact match for incoming packets to a previous outgoing TCP packet – addresses, ports, TCP sequence numbers – before they allow a connection to be established

Connection Establishment in a Fragmented Network

- Client-server connection establishment
- Impact of TLS and IPv6 on connection establishment
- Peer-to-peer connections, NATs, and NAT traversal