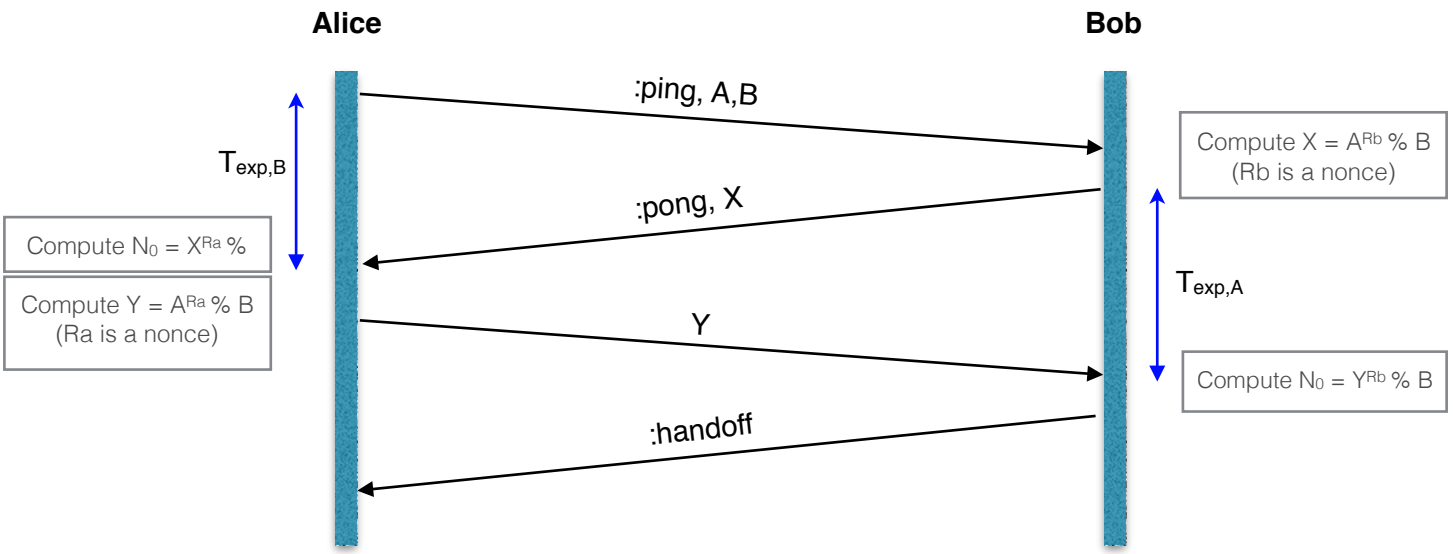
Connection Setup

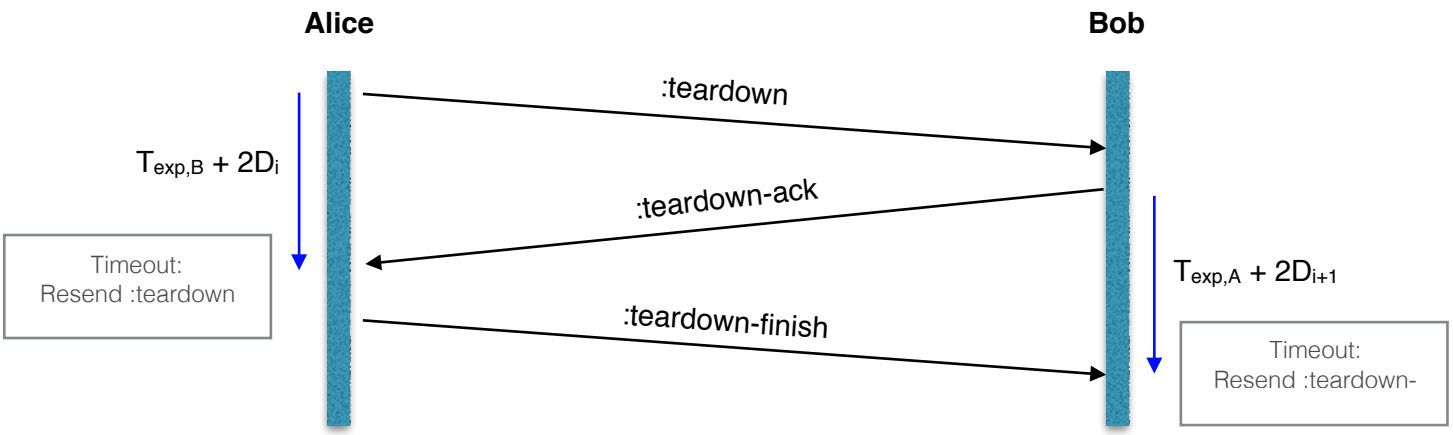
Precondition: Alice and Bob share shared secret key K_0
Postcondition: Alice and Bob share K_0 and N_0. Alice knows $T_{exp,B}$ and Bob knows $T_{exp,A}$



I. Alice initiates a communication with Bob by sending Diffie-Hellman generator values A and B along with a simple :ping message
II. Bob computes X (per the Diffie-Hellman exchange), and returns it to Alice along with a simple :pong message
III. Alice measures $T_{exp,B}$, the time she expects it to take to receive a response from Bob, and computes N_0 and Y. She then sends Y to Bob
IV. Bob computes N_0 and sends a :handoff message to Alice

Connection Teardown

Precondition: Alice and Bob share a connection
Postcondition: Alice and Bob have both freed the resources related to their connection

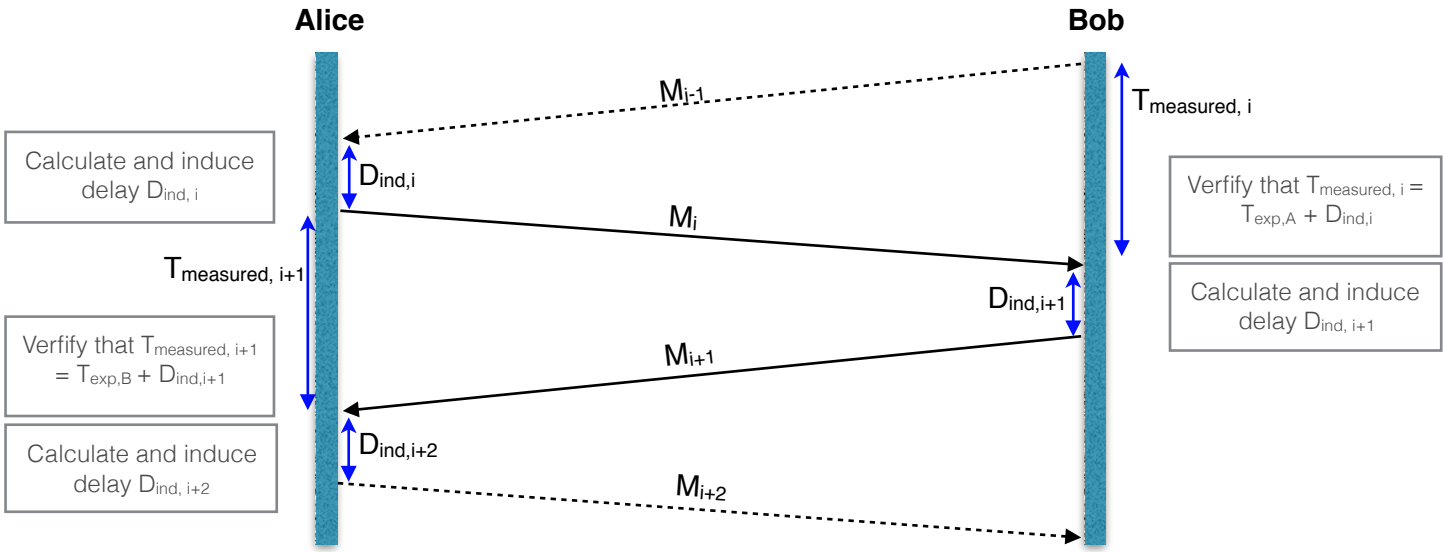


Same teardown policy as TCP.

- Alice initiates teardown by sending :teardown
- Bob Acknowledges teardown by sending :teardown-ack
- If Alice doesn't receive :teardown-ack in expected timeframe, she resends :teardown and releases the connection locally. Otherwise, she sends :teardown-finish
- If Bob doesn't receive :teardown-finish in expected timeframe, he resents :teardown-ack. He releases the connection locally when he receives :teardown-finish or when a timeout occurs

Standard Communication

Precondition: Alice and Bob share a secret key stream N generated with a shared secret key K



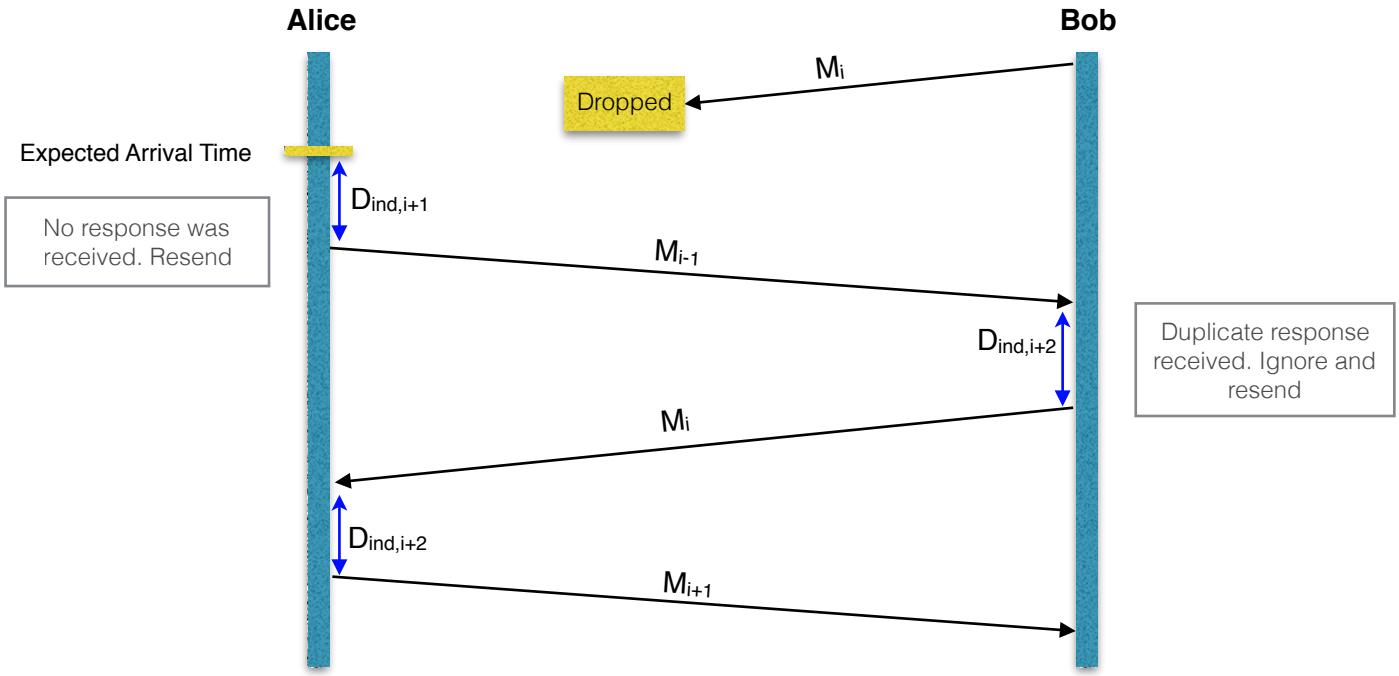
Sender
Before send any message indexed i in the communications between Alice and Bob, the sending party artificially induces a delay $D_{ind,,i}$. If they have nothing to send at the end of the delay, they send a :handoff message.

Recipient
Upon receiving a message, the recipient verifies that the time it took to receive the response equals the expected response time of the other party plus the induced delay ($T_{exp,sender} + D_{ind,\ i}$).

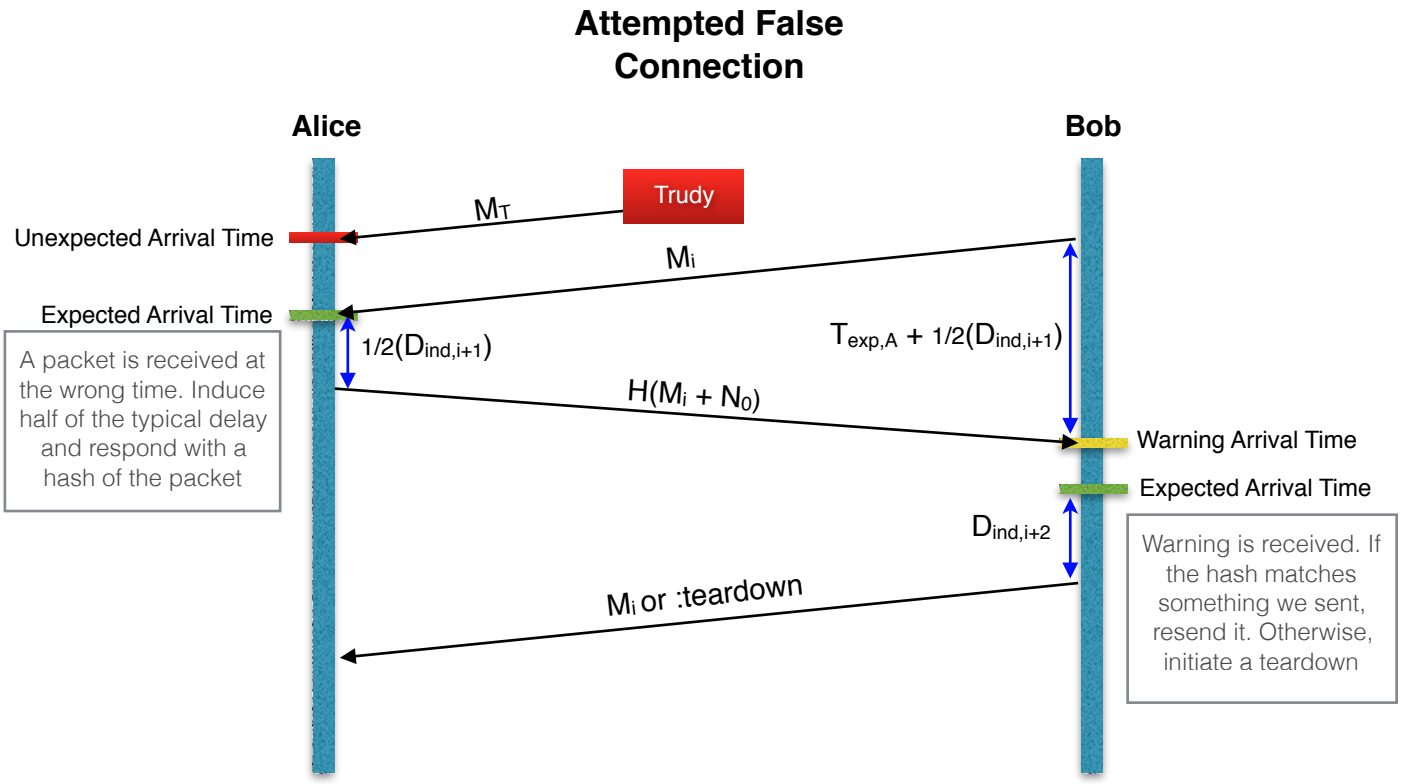
Computing $D_{ind,i}$
$D_{ind,i}$ is computed by extracting the next M bits of the key stream generated with the stream cipher initialized with the shared secret key K and initialization vector N_0.

For a key stream KS, these bits are $KS_{M*i\ \ldots\ M*(i+1)}$ for a zero-indexed i.

Dropped Packet



If Alice does not receive a response from Bob when she expects one, she will induce the next delay and then re-send her previous packet. If the original packet from Alice to Bob was dropped, then she resends that packet. If the response from Bob to Alice was lost, then Alice sends a duplicate message (unknowingly). Bob, upon receiving this, will retransmit his response to Alice upon receiving the new packet and inducing the subsequent delay.



If Alice receives a packet at an incorrect time, she induces half of the typical delay and sends a response to Bob with the hash of the received message and N_0. When Bob receives the warning message, he can tell that Alice received a packet at the wrong time because he received the packet after half of the expected delay. If the hash Alice sent matches a message he sent, he can re-send that message, because he knows that the message Alice received was, in fact, his. Otherwise, he can initialize a teardown sequence.

If multiple unexpected arrivals occur in a small window of time, and all turn out to be from the actual sender, the initialization sequence will be restarted to adjust expected response times.