Algorithm 3 Send and receive one message on a client.

Input: $state, e, r, successful, isSecondTx$

if $successful \wedge \neg isSecondTx$ **then**
  $payload \leftarrow$ accept next *message* from *user* or generate *cover message*
end if
$successful \leftarrow \bot$
while $(\neg successful) \wedge (\text{time.Now}() < r.ClosingTime)$ **do**
  for $c \leftarrow 1 \ldots state.CascadesMatrix^{(e)}$ **do** ➤ Prepare ephemeral keys for each mix.
    for $m \leftarrow 1 \ldots state.CascadesMatrix^{(e)}[c]$ **do**
      $keys[c][m].pk, keys[c][m].sk \leftarrow$ generate message-only key pair
      $keys[c][m].k \leftarrow \text{DH}(state.CascadesMatrix^{(e)}[c][m].pk, keys[c][m].sk)$
    end for
  end for
  for $c \leftarrow 1 \ldots state.CascadesMatrix^{(e)}$ **do** ➤ Pad, onion-encrypt, and send to each cascade.
    $sendMsg.pk, sendMsg.payload \leftarrow \text{pad}(state.Peer^{(e)}), \text{pad}(payload)$
    for $m \leftarrow state.CascadesMatrix^{(e)}[c] \ldots 1$ **do**
      $sendMsg.pk, sendMsg.payload \leftarrow keys[c][m].pk, enc_{keys[c][m].k}(sendMsg.pk \mid\mid sendMsg.payload)$
    end for
    $successful \leftarrow$ send $sendMsg$ to entry of $state.CascadesMatrix^{(e)}[c]$ ➤ Possibly including FEC data.
  end for
end while
if $successful$ **then**
  if $isSecondTx$ **then**
    $isSecondTx \leftarrow \bot$
  else
    $isSecondTx \leftarrow \top$
  end if
end if
for $c \leftarrow 1 \ldots state.CascadesMatrix^{(e)}$ **do**
  $recvMsg \leftarrow$ receive from exit of $state.CascadesMatrix^{(e)}[c]$ ➤ Possibly corrected via FEC data.
  if $\neg\text{isCoverMsg}(recvMsg) \wedge \neg\text{alreadySeen}(e, r, recvMsg)$ **then**
    yield $recvMsg$ to application layer
  end if
end for

return $successful, isSecondTx$