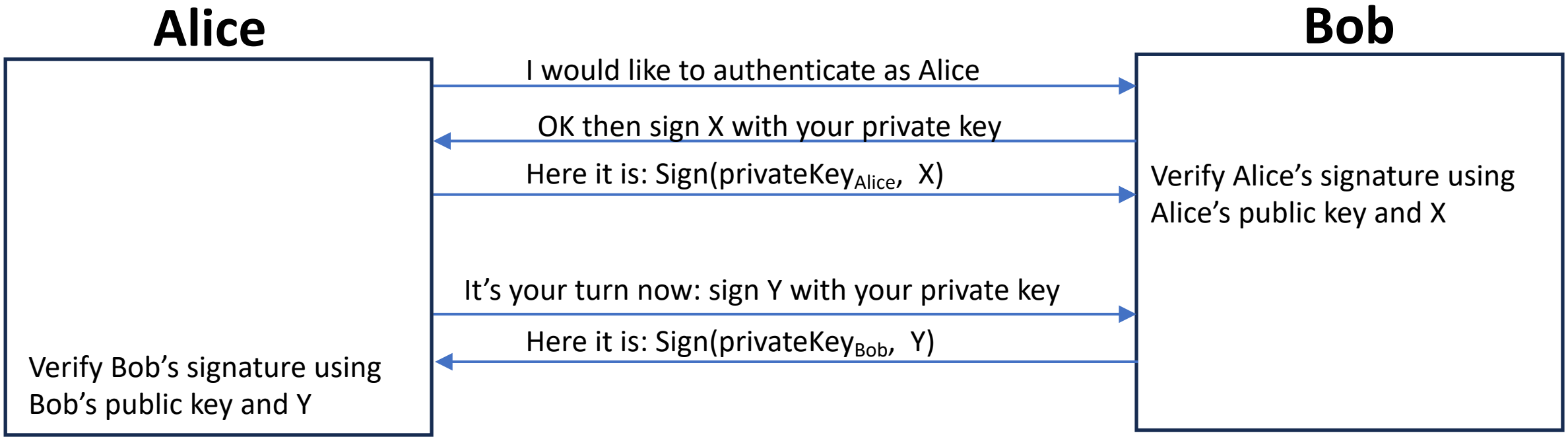
# Challenge/Response protocol using Java RMI

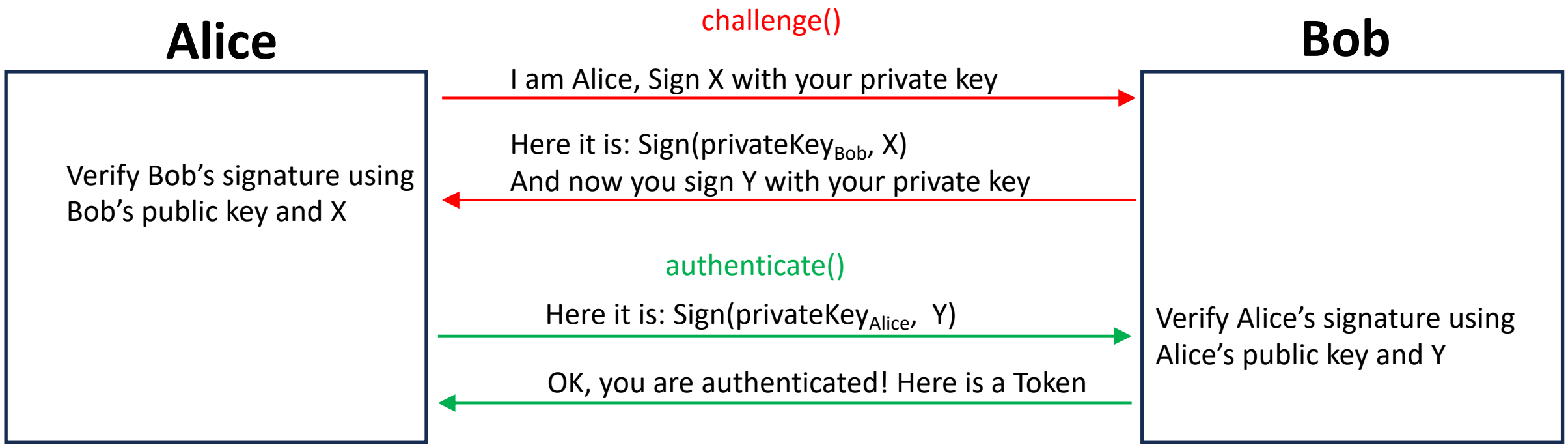
# Challenge/Response Protocol (a key-based Authentication Protocol)



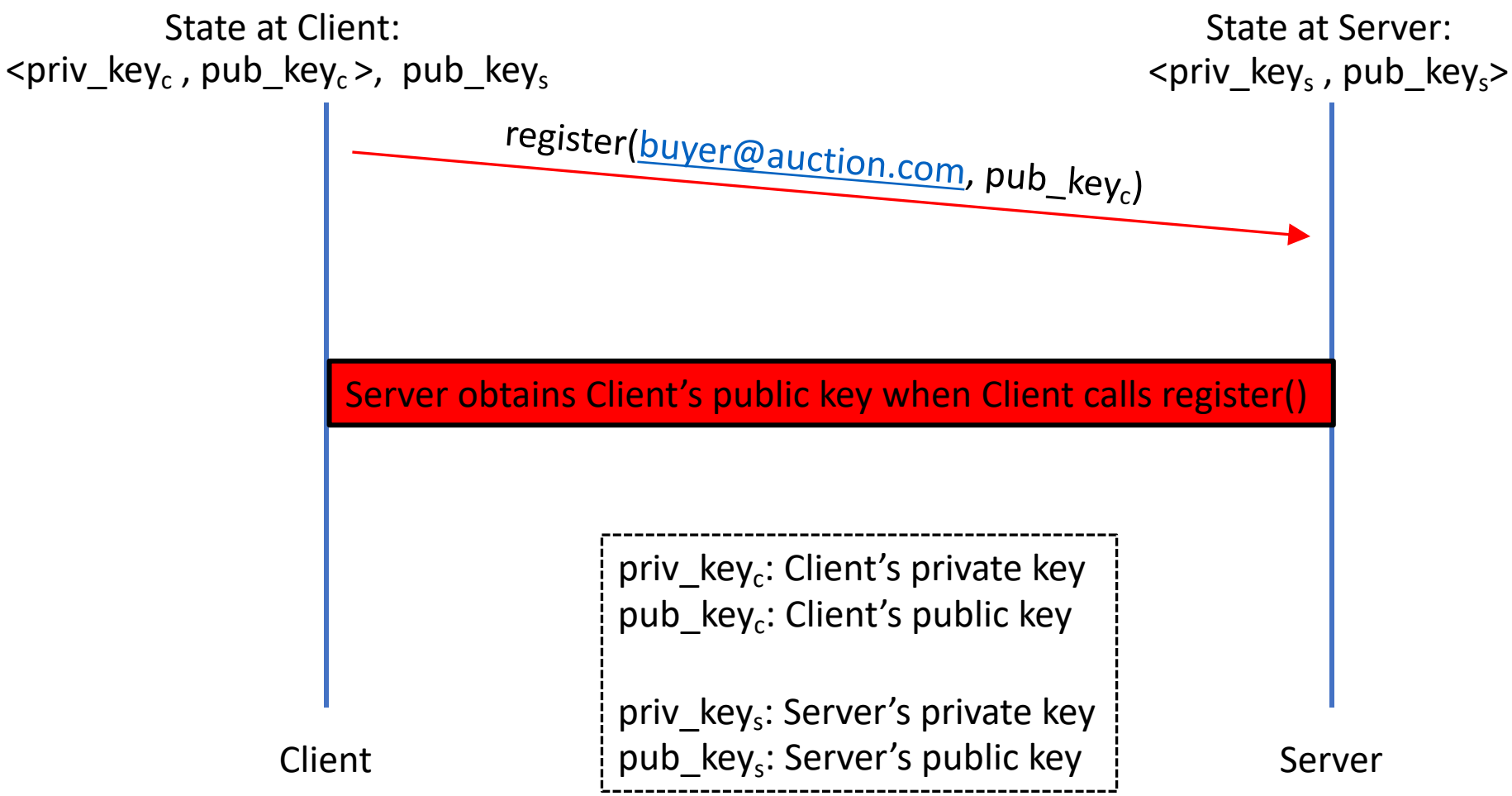

- X and Y are (random) challenges

# Coursework

- A slight variant of the challenge/response protocol
- The Client authenticates the Server first (as part of challenge() call)
- Afterwards, the Server authenticates the Client (as part of authenticate() call)



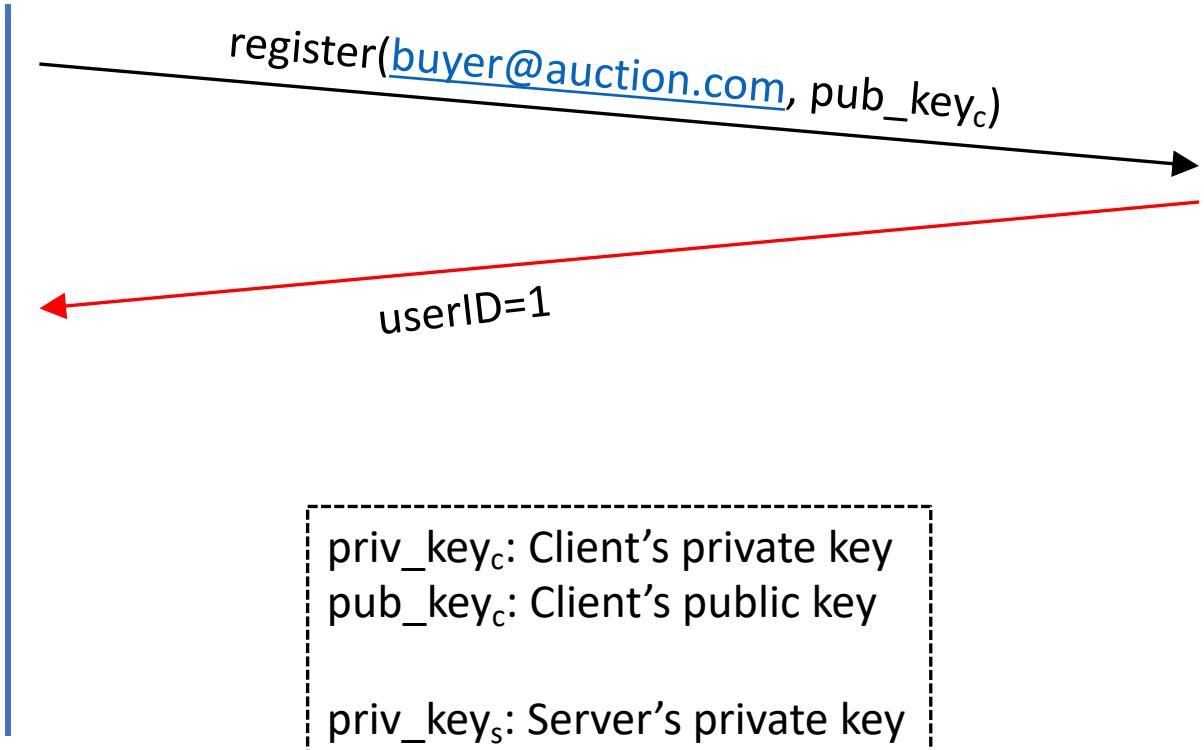
# Coursework


State at Client:
<priv_key_c , pub_key_c >, pub_key_s
State at Server:
<priv_key_s , pub_key_s>
register(buyer@auction.com, pub_key_c)
Server obtains Client's public key when Client calls register()
priv_key_c: Client's private key
pub_key_c: Client's public key
priv_key_s: Server's private key
pub_key_s: Server's public key
Client
Server

# Coursework

State at Client:
<priv_key$_c$ , pub_key$_c$ >, pub_key$_s$

State at Server:
<priv_key$_s$ , pub_key$_s$>, pub_key$_c$



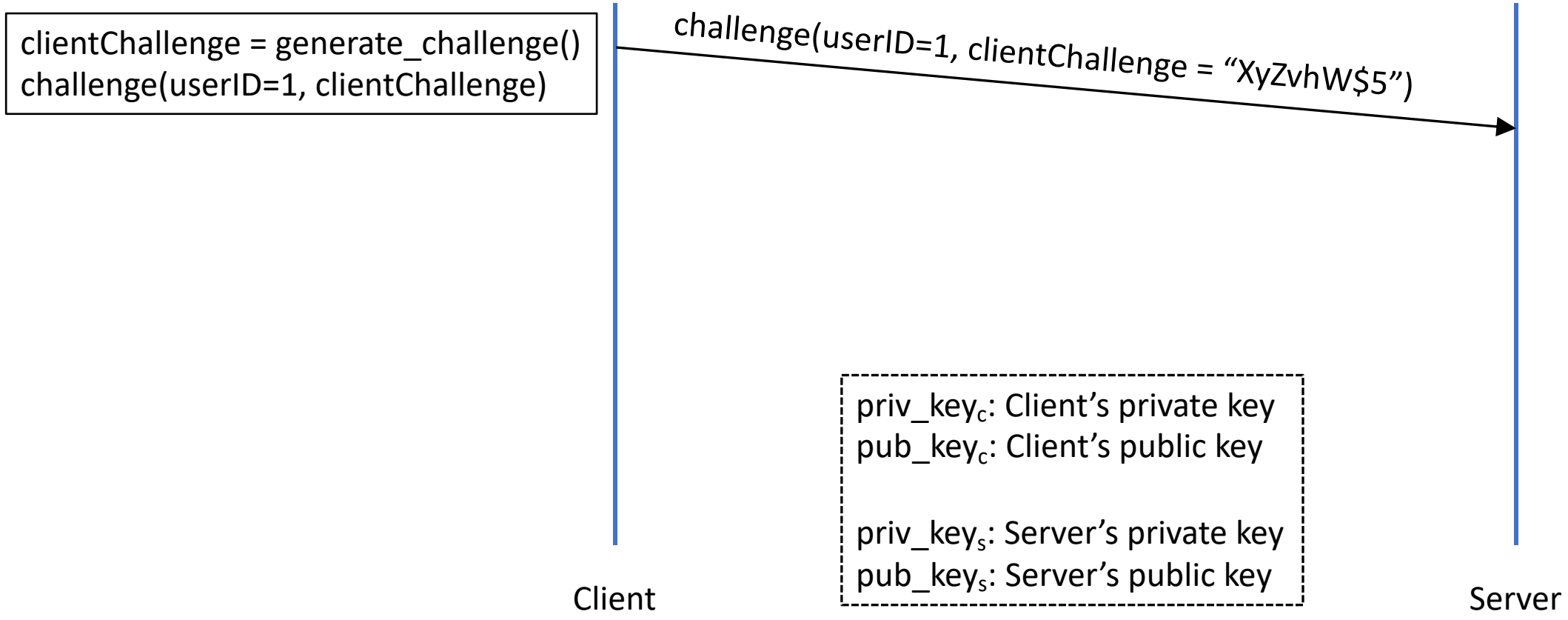

priv_key$_c$: Client's private key
pub_key$_c$: Client's public key

priv_key$_s$: Server's private key
pub_key$_s$: Server's public key

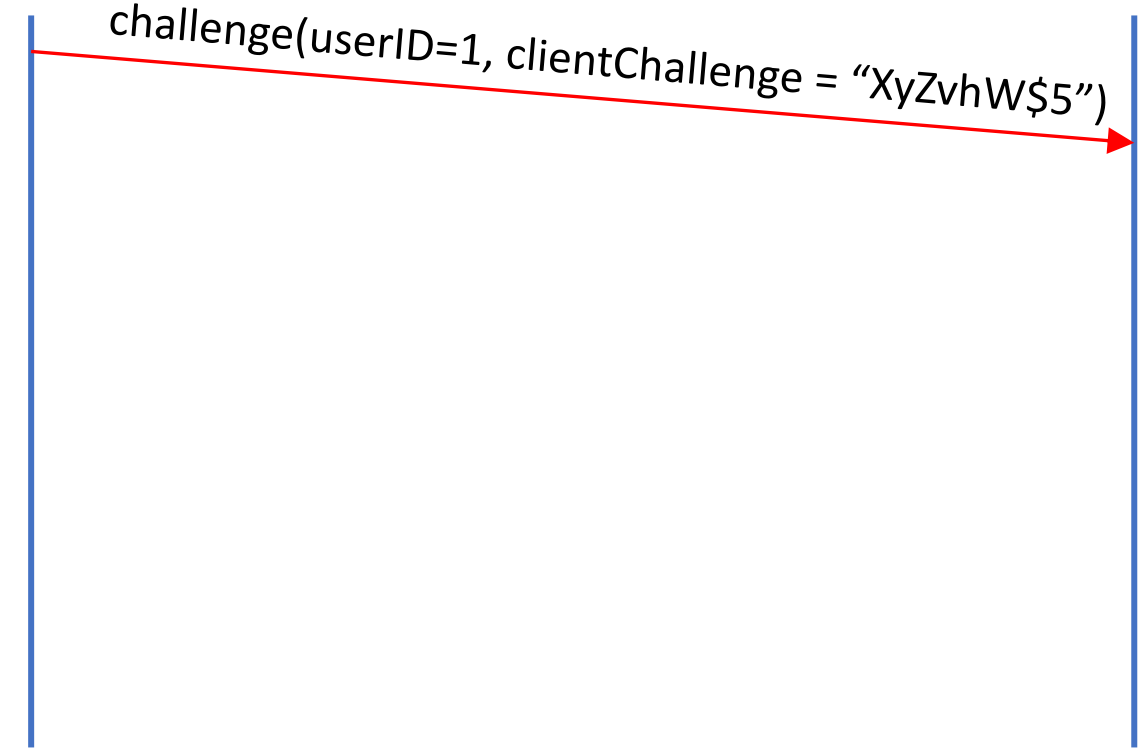
Client

Server

# Coursework


clientChallenge = generate_challenge()
challenge(userID=1, clientChallenge)
challenge(userID=1, clientChallenge = "XyZvhW$5")
priv_key$_c$: Client's private key
pub_key$_c$: Client's public key
priv_key$_s$: Server's private key
pub_key$_s$: Server's public key
Client
Server

# Coursework

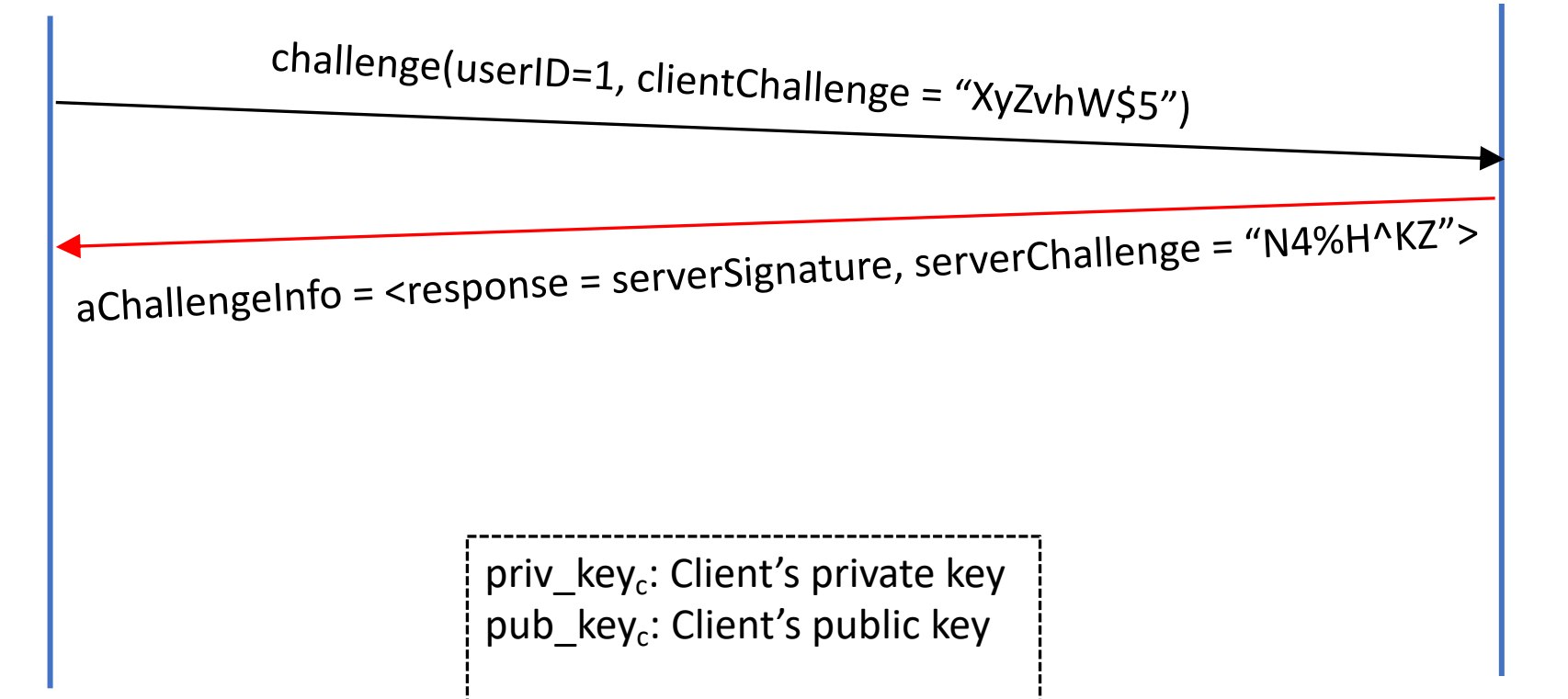



serverSignature = Sign("XyZvhW$5") using $priv\_key_s$
serverChallenge = generate_challenge()
challengeMap.put (<userID=1, serverChallenge>)
return new ChallengeInfo (serverSignature,
serverChallenge="N4%H^KZ")

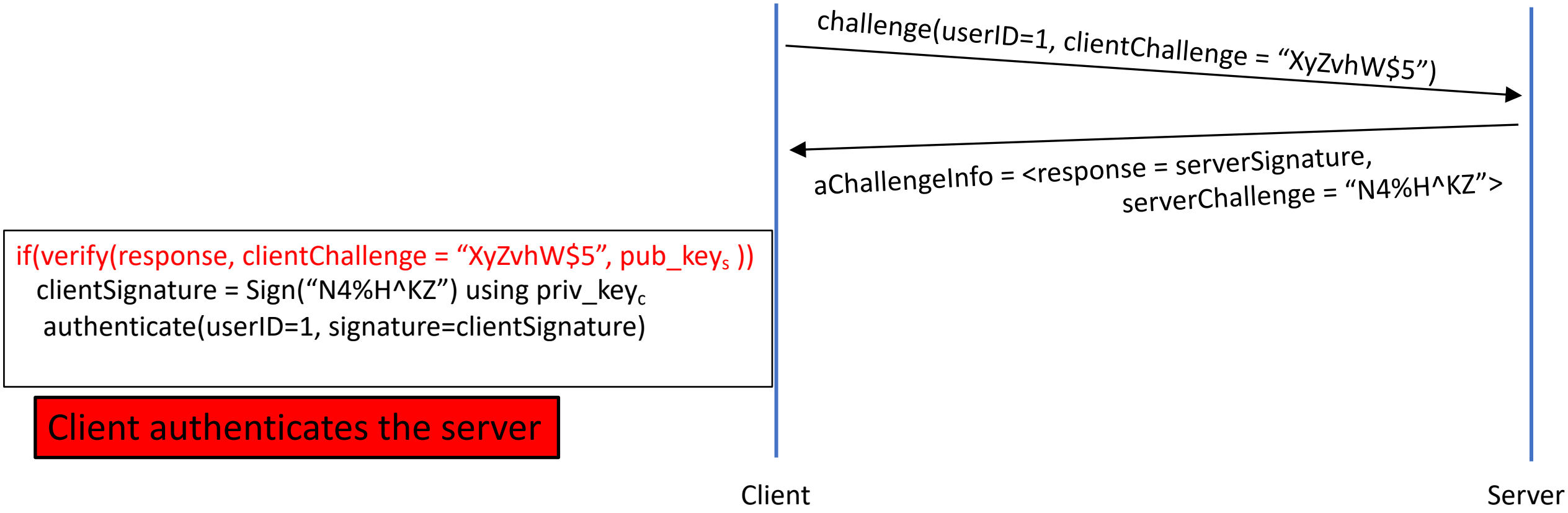
$priv\_key_c$: Client's private key
$pub\_key_c$: Client's public key

$priv\_key_s$: Server's private key
$pub\_key_s$: Server's public key

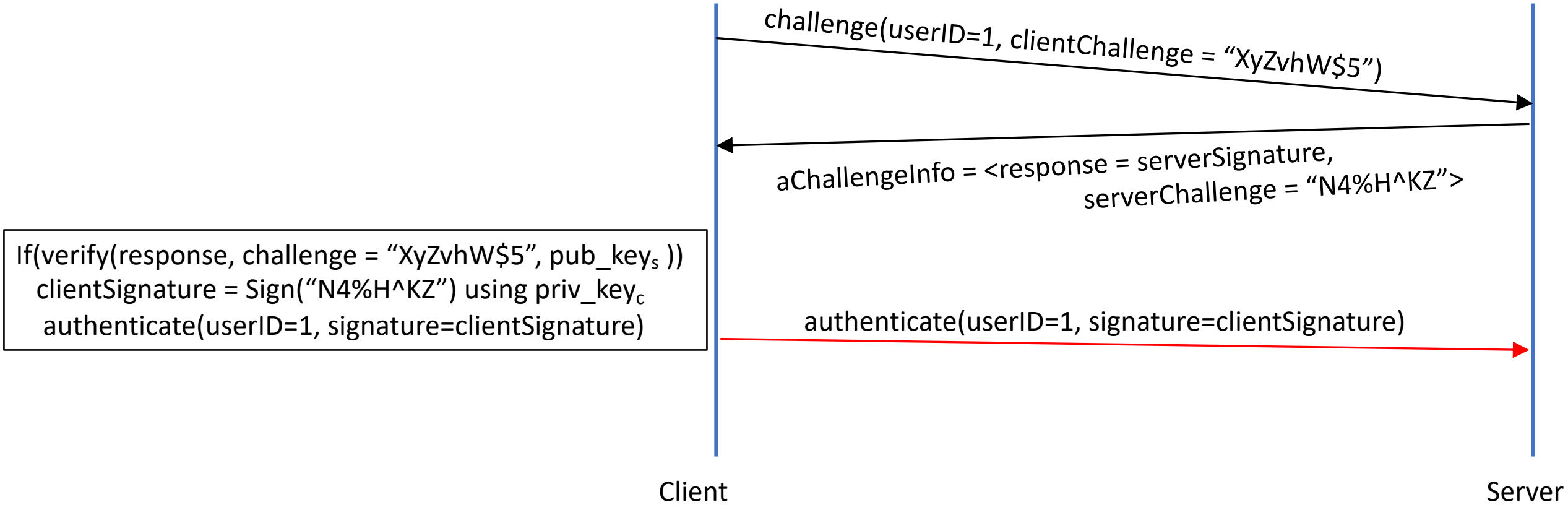
# Coursework


challenge(userID=1, clientChallenge = “XyZvhW$5”)
aChallengeInfo = <response = serverSignature, serverChallenge = “N4%H^KZ”>
Client
Server
priv_key$_c$: Client’s private key
pub_key$_c$: Client’s public key
priv_key$_s$: Server’s private key
pub_key$_s$: Server’s public key

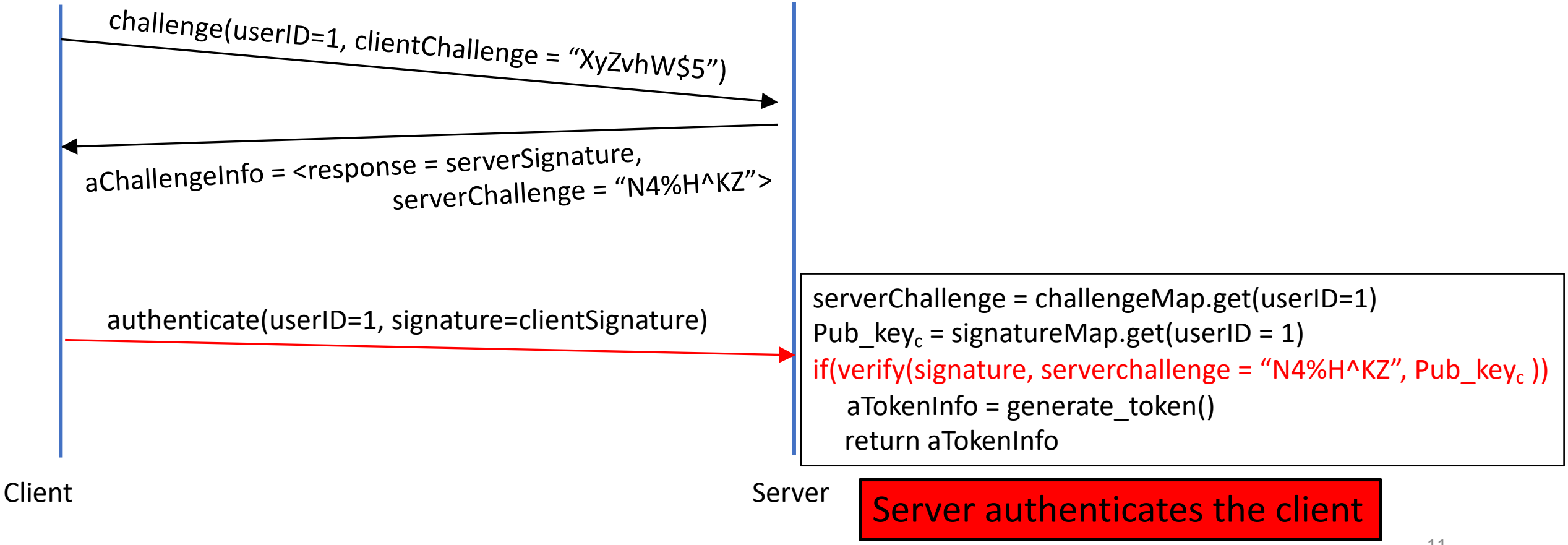
# Coursework


challenge(userID=1, clientChallenge = “XyZvhW$5”)
aChallengeInfo = <response = serverSignature,
serverChallenge = “N4%H^KZ”>
if(verify(response, clientChallenge = “XyZvhW$5”, pub_key$_s$ ))
clientSignature = Sign(“N4%H^KZ”) using priv_key$_c$
authenticate(userID=1, signature=clientSignature)
Client authenticates the server
Client
Server

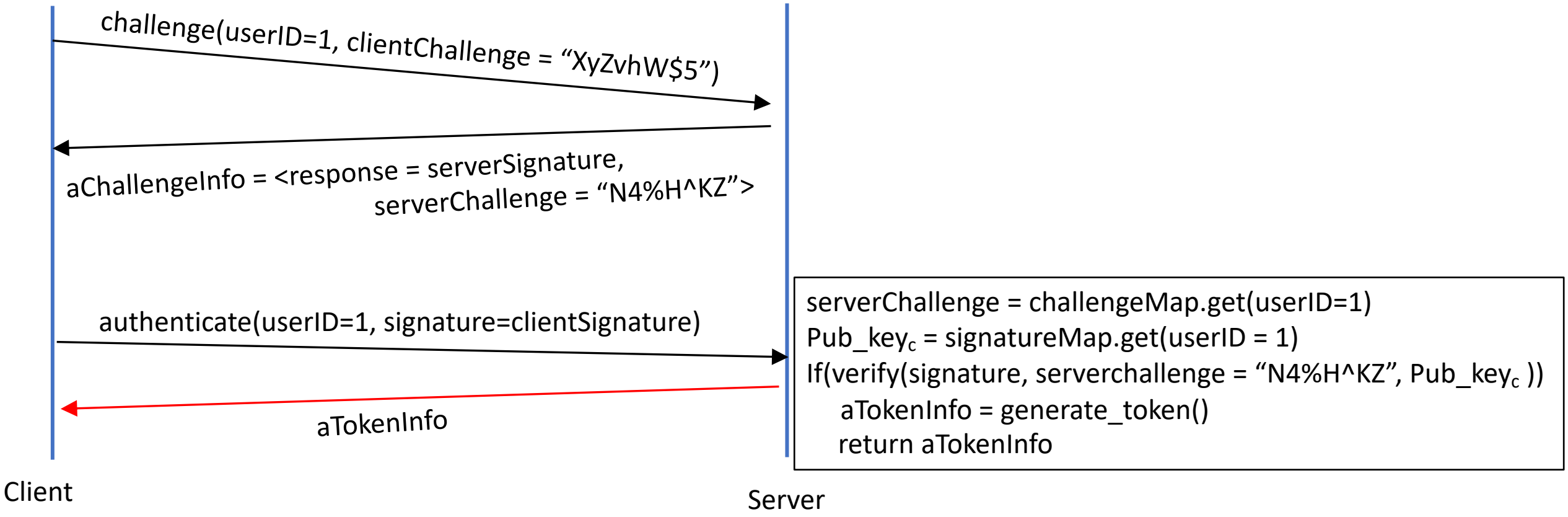
# Coursework


challenge(userID=1, clientChallenge = “XyZvhW$5”)
aChallengeInfo = <response = serverSignature,
serverChallenge = “N4%H^KZ”>
If(verify(response, challenge = “XyZvhW$5”, pub_key$_s$ ))
clientSignature = Sign(“N4%H^KZ”) using priv_key$_c$
authenticate(userID=1, signature=clientSignature)
authenticate(userID=1, signature=clientSignature)
Client
Server

# Coursework


challenge(userID=1, clientChallenge = "XyZvhW$5")
aChallengeInfo = <response = serverSignature,
serverChallenge = "N4%H^KZ">
authenticate(userID=1, signature=clientSignature)
serverChallenge = challengeMap.get(userID=1)
Pub_key$_c$ = signatureMap.get(userID = 1)
if(verify(signature, serverchallenge = "N4%H^KZ", Pub_key$_c$ ))
aTokenInfo = generate_token()
return aTokenInfo
Client
Server
Server authenticates the client

# Coursework


challenge(userID=1, clientChallenge = "XyZvhW$5")
aChallengeInfo = <response = serverSignature,
serverChallenge = "N4%H^KZ">
authenticate(userID=1, signature=clientSignature)
aTokenInfo
serverChallenge = challengeMap.get(userID=1)
Pub_key$_c$ = signatureMap.get(userID = 1)
If(verify(signature, serverchallenge = "N4%H^KZ", Pub_key$_c$ ))
aTokenInfo = generate_token()
return aTokenInfo
Client
Server