Exercise 16

Sunday, 19 February 2023 18:12

a)

$Send(net, A, k_{as}) ==$ out net A; inp net x; new s; out net $\{s, x\}_{k_{as}}$;

$Recv(net, k_{bs}) ==$ inp net x; new n; out net n; inp net y; out net $\{x, y\}_{k_{bs}}$; inp net z; decrypt z is $\{z_1\}_{k_{bs}}$; if $x \neq z_1$ then secret(z_1)

$Server(net, A, k_{as}, k_{bs}) ==$ inp net x; decrypt x is $\{x_1\}_{k_{bs}}$; split x_1 is (y_1, y_2); if $y_1 = A$ then decrypt y_2 is $\{z\}_{k_{as}}$; out net $\{z\}_{k_{bs}}$;

Compromised(k_{spy}) == out net k_{spy};

$P(net, A, B, spy) ==$ new k_{spy}, k_{as}, k_{bs}; !Send(net, A, k_{as}) | !Send(net, B, k_{bs}) | !Recv(net, k_{bs}) | !Recv(net, k_{as}) | !Server(net, A, k_{as}, k_{bs}) | !Server(net, B, k_{bs}, k_{as}) | !Server(net, A, k_{as}, k_{spy}) | !Server(net, B, k_{bs}, k_{spy}) | !Server(net, spy, k_{spy}, k_{as}) | !Server(net, spy, k_{spy}, k_{bs})

b)

This answer is a bit different from the one in the solutions. The main idea of this answer is to simply pass the challenge from the spy to the server, then have the server present the decrypted secret (which was intended for B) to the spy. However, this solution has one problem: it assumes that the server accepts that the same nonce is seen twice. Strictly speaking, the protocol for the server does not check this. However, if this were to occur in reality, the opponent process below would not work.

0 == inp net k_{spy}; get k_{spy} from compromised

inp net n_{BA}; get nonce sent from B→A

out net n_{BA}; make sure this nonce can still be used by A and B.

inp net $sn_{k_{as}}$; get encrypted message containing s from A
out net $sn_{k_{as}}$; immediately forward it to B

out net $\{(a, sn_{k_{as}})\}_{k_{spy}}$; forward challenge to server, but with spy's key

inp net $s_{k_{spy}}$; server answers with $\{s\}_{k_{spy}}$

decrypt $s_{k_{spy}}$ is $\{s\}_{k_{spy}}$; decrypt

out net s; output to net → s is not secret anymore

Alternatively, we could consider the following opponent process, which is based on the solutions.

0 == inp net k_{spy}; get spy's key

out net A;
inp net n_A; } Create session O(A) → B

out net spy;
inp net n_{spy}; } Create session O(spy) → B

new g; fake $\{s_A, n_A\}$ for session O(A) → B

new s_{spy}; s for O(spy) → B which B will eventually misinterpret as being for O(A) → B

out net g; have B send $\{A, g\}$ to S note: B thinks this is the message $\{A, \{s_A, n_A\}_{k_{as}}\}_{k_{bs}}$

B will not get a response, due to the message being malformed

out net $\{(s_{spy}, n_A)\}_{k_{spy}}$; B will forward this message as

$\{(spy, \{(s_{spy}, n_A)\}_{k_{spy}})\}_{k_{bs}}$

and receives back

$\{s_{spy}\}_{k_{bs}}$

which he misinterprets as the response to $\{A, \{s_A, n_A\}_{k_{as}}\}_{k_{bs}}$
and thus, the secret s_{spy} is misinterpreted as being shared with A.

out net s_{spy}; O publishes s_{spy}, violating its secrecy