```
Resp(net,B,A, pA) ==    inp net x; split x is (m, a);
                        if a = A then
                        new n; out net n;
                        inp net y; decrypt y is {|z|}pA;
                        if z = #(m,B, n) then stop
```

global
local

in spi, net is a free name

free names model public data

---

finish protocols

$Send(A,B,net,k) == new\ s,p,s',p'; out\ net\ (\{s,p\}_k, \{p',s'\}_k);$

$Recv(A,B,net,k) == inp\ net\ x; split\ x\ is\ (x_1,x_2);\ decrypt\ x_1\ is\ \{y\}_k;\ decrypt\ x_2\ is\ \{y\}_k$

another way to fix the protocols is to include 'type tags' in the encrypted messages

assertion

$A ::== secret(M)$ M is secret

a process Q is called an error state iff it has the following form

$Q = new\ n_1,\dots n_j; (secret(M) | out\ c\ M; Q' | Q'')$

where $c \notin n_1,\dots,n_j$

an error state occurs when a secret is sent to a public channel

a closed process P is safe for secrecy iff $P \longrightarrow^* Q$ implies that Q is not an error state

an opponent process is a closed process that does not contain assertions

a closed process P is robustly safe for secrecy iff for all opponent processes O the parallel composition P|O is safe for secrecy

if a process is not robustly safe for secrecy, a proper algorithm will actually find 'a bug'/'a reason'.
due to undecidability, however, the algorithm may go on forever...
a process which are robustly safe for secrecy

external threat model: assumes that all regular protocol participants are honest
internal threat model: has compromised protocol participants as well

for internal threats, you want to check that secrecy holds with respect to all non-compromised participants

internal threats are modeled by publishing the secrets of one of the agents
↓
which is called the spy

conditional assertions: if $M \neq N$ then A ("Provided $M \neq N$")

results in $Receive(a,b,k) = inp\ net;\ decrypt\ x\ is\ \{y\}_k;\ if\ a \neq b\ then\ secret(y)$