Intercept and Resend Attack Modeling

Version: Adversary has access to polarization basis chosen by encoder

```
% Rectilinear & diagonal basis angles
basis=[[90, 0] ; [45, 135]];

% Random data bits
Data=randi([0 1],1,10);

% Encoding
[S,polarization]=encoder(basis,Data);

% Insecure Quantum channel
[R,DC]=channel(basis,polarization,S);

% Decoding
[D,basisindex]=decoder( basis,R );
```

```
% Authenticated channel handshake
indices=authenticatedChannelHandshake(
    polarization,basisindex);

% Keep only bits measured using same basis
D=D(indices); % D is the sifted key

% Compare Alice's random data bits with sifted
key
if Data(indices)==D
    fprintf('Key established with success!\n');
else
    fprintf('Key establishment failed!\n');

% Compare Eve's intercepted bits with sifted key
if DC(indices)==D
    fprintf('Key intercepted with success!\n');
else
    fprintf('Key interception failed!\n');
```

Intercept and resend attack

```
% S = photon angle sequence
% Decoded bits
D=[];
% Resent photons
R=[];
for i=1:length(S)
    % Intercept
    if S(i)==basis(polarization(i),1)
        D=[D, 0]; % binary zero!
    else
        D=[D, 1]; % binary one!
    end
    % Resend
    % angles in corresponding basis
    A=basis(polarization(i),:);
    % map data bit to photon
    R=[R A(D(i)+1)];
```

Example

```
Data (Alice):
   1  0  0    1    1    0    0    0    0    0
Angles(S):
 135 45  45   0 135   90   90   90   90   90
Polarization:
   2  2   2   1    2    1    1    1    1    1

D (Bob):
   0  0   0   1    0    0    0    0    0    0
DC (Eve):
   1  0   0   1    1    0    0    0    0    0

Basisindex:  1

Indices:  4  6  7  8  9  10

D (sifted key):  1  0  0  0  0  0

Key established with success!
Key intercepted with success!
```

Intercept and Resend Attack Modeling

Version: Adversary has no access to polarization basis chosen by encoder

```
% rectilinear & diagonal basis angles
basis=[[90, 0] ; [45, 135]];
% Random data bits
Data=randi([0 1],1,L);

% Encoding
[S,polarization]=encoder(basis,Data);

% Insecure Quantum channel
[R,DC]=channel(basis,S);

% Decoding
[D,basisindex]=decoder(basis,R);
```

```
% Authenticated channel handshake
indices=authenticatedChannelHandshake(
   polarization,basisindex);
% keep bits measured using same basis
D=D(indices); % D is the sifted key

If % Compare Alice's bits with sifted key
   Data(indices)==D and
   % Compare Eve's bits with sifted key
   DC(indices)==D
   print('Both key establishment and interception success.');
```

Intercept and resend attack

```
% S = photon angle sequence
% Decoded bits
D=[];
% Resent photons
R=[];
for i=1:length(S)
    % Intercept
    % randomly select a basis
    A=basis(randi([1 2]),:);
    % Apply the basis
    if S(i)==A(1)
        D=[D,0]; % binary zero!
    elseif S(i)==A(2)
        D=[D,1]; % binary one!
    else
        D=[D,randi([0 1])]; % random bit!
    end
    % Resend decoded bit as a new photon
    R=[R A(D(i)+1) ];
```

Example

Data (Alice):

0 1 0 0 0 0 0 0 1 0

Angles(S):
90 135 90 90 90 45 45 45 135 90

Polarization:

1 2 1 1 1 2 2 2 2 1

D (Bob):

0 1 0 0 0 1 0 1 1 1

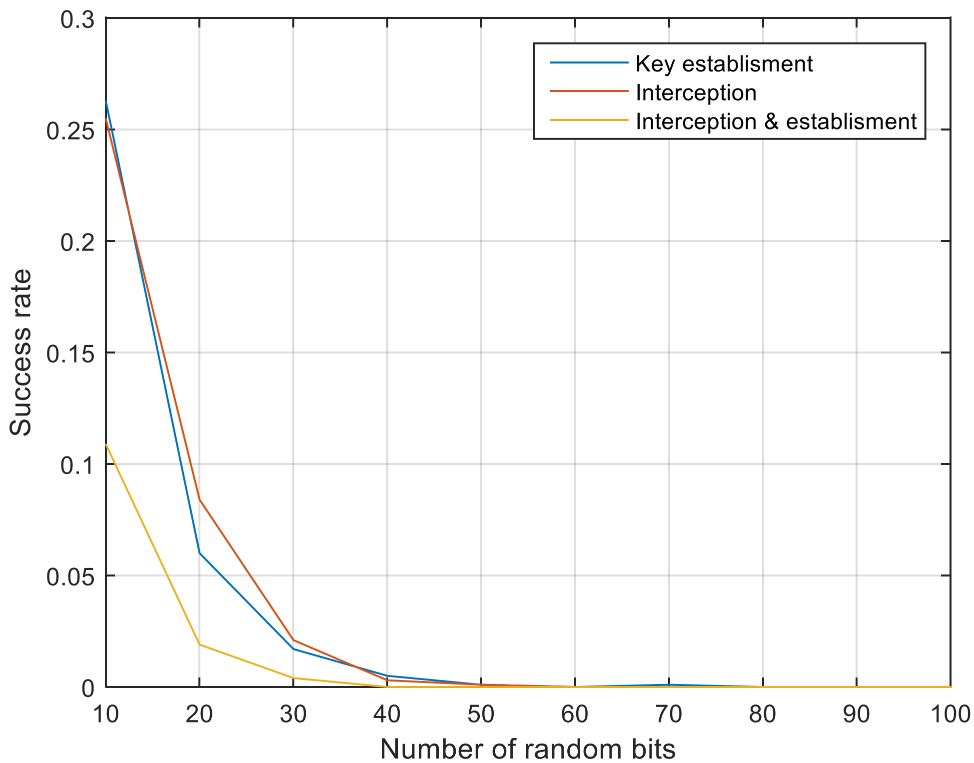
DC (Eve):

0 1 1 1 0 0 0 1 1 0

Basisindex: 1

Indices: 1 3 4 5 10

D (sifted key): 0 0 0 0 1


Key establisment
Interception
Interception & establisment
Success rate
0
0.05
0.1
0.15
0.2
0.25
0.3
10
20
30
40
50
60
70
80
90
100
Number of random bits