日本語訳『Qiskit Textbook』勉強会 第3章（3.12）

Shun Shirakawa

Application Architect

自己紹介

名前：白川　俊

所属：IJDS デジタル事業部

仕事：

- Application Architect
- 製造業のお客様向けのシステム構築
- 最適化

History

- CSC（1992～）
  - CADパッケージソフト開発
- IASC（2004～）
  - PLM　設計システム構築
- ISC-J（2010～）
  - SCM　生産システム構築
  - Analytics
- IJDS（2020～）

3.12

量子鍵配送

Quantum Key Distribution

目次

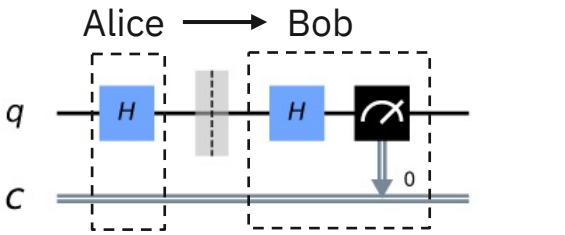
1. はじめに
2. プロトコル概要
3. Qiskitの例（盗聴なし）
4. Qiskitの例（盗聴あり）
5. リスク分析

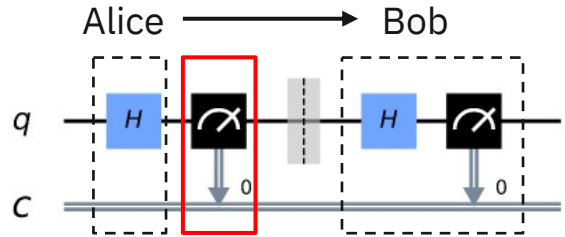
1. はじめに

- 安全でない通信路上での通信には、メッセージの暗号化が必要
  - 秘密鍵があれば共通鍵暗号で安全に通信できる、という前提
- 鍵の共有手段としての通信路
  - 古典的通信路→盗聴されているかどうかを知ることは不可能
  - 量子的通信路→盗聴しようとするとそのことが発覚する
- 物理的実装のイメージ
  - 古典的通信路：電話線
    - 電気信号でビットを表現し送信
  - 量子的通信路：光ファイバーケーブル
    - 光子の *偏光* で量子ビットを表現し送信

2. プロトコルの概要

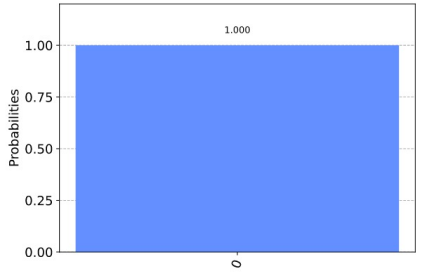
- 「量子ビットを測定すると、状態が変化しうる」ことを利用

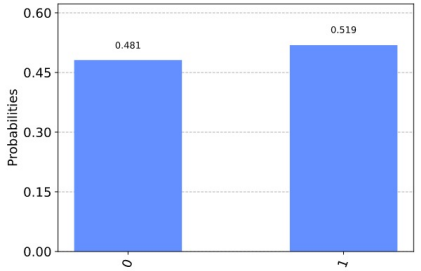




Eveが盗聴

Bobの測定結果





2. プロトコルの概要

2. プロトコルの概要

Alice

1000101011010100

①ランダムなビット列を選ぶ
ランダムな基底列を選ぶ

ZZXZXXXZXZXXXXXX

2. プロトコルの概要

Alice

②量子ビット上に
エンコードして送信

1000101011010100

①ランダムなビット列を選ぶ
ランダムな基底列を選ぶ

ZZXZXXXZXZXXXXXX

	0	1
Z	|0>	|1>
X	|+>	|->

$|1\rangle|0\rangle|+\rangle|0\rangle|-\rangle|+\rangle|-\rangle|0\rangle|-\rangle|1\rangle|+\rangle|-\rangle|+\rangle|-\rangle|+\rangle|+\rangle$

$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \quad |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$

テキストではこうなっているが…

$|-\rangle|0\rangle|+\rangle|0\rangle|1\rangle|0\rangle|1\rangle|+\rangle|1\rangle|-\rangle|+\rangle|-\rangle|0\rangle|-\rangle|0\rangle|+\rangle$

2. プロトコルの概要

Alice

1000101011010100

①ランダムなビット列を選ぶ
ランダムな基底列を選ぶ

②量子ビット上に
エンコードして送信

ZZXZXXXZXZXXXXXX

	0	1
Z	|0>	|1>
X	|+>	|->

|1>|0>|+>|0>|−>|+>|−>|0>|−>|1>|+>|−>|+>|−>|+>|+>

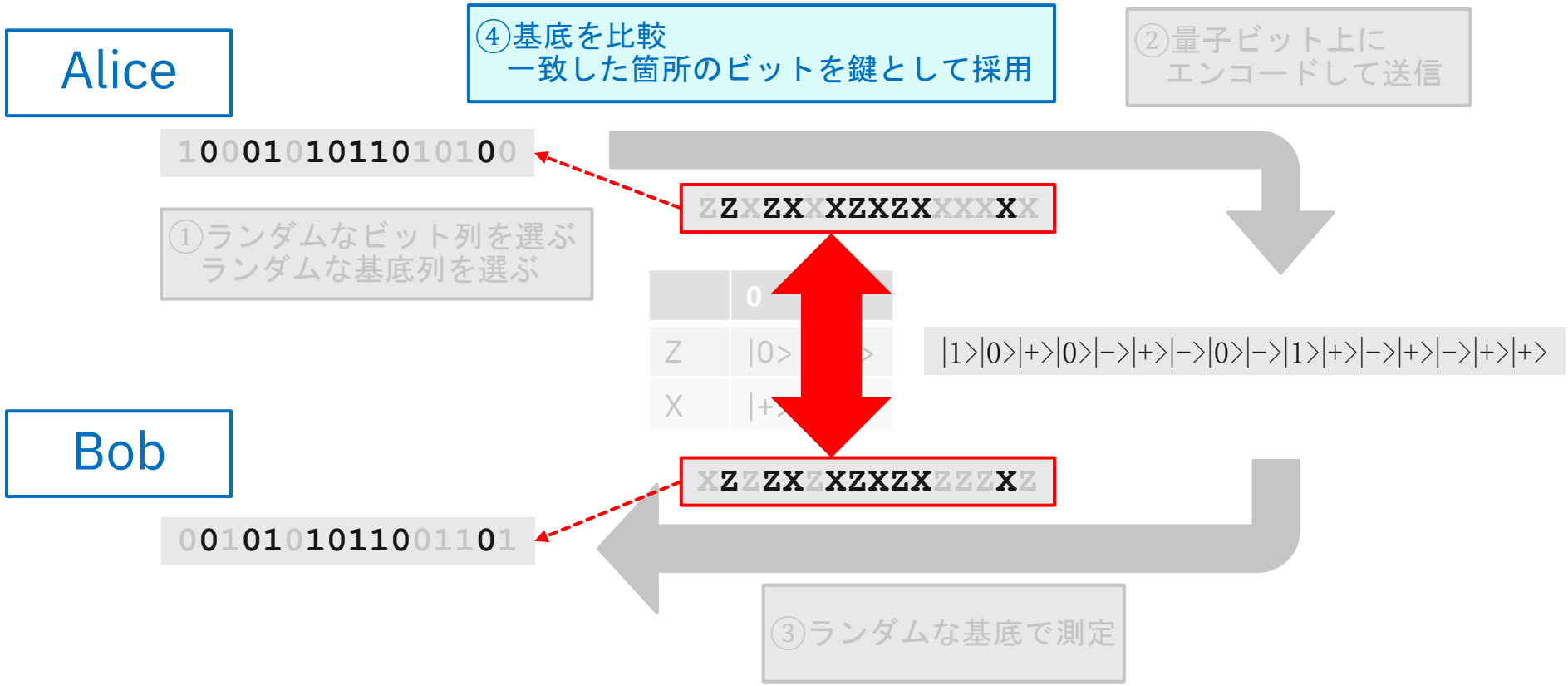
Bob

001010101100l101

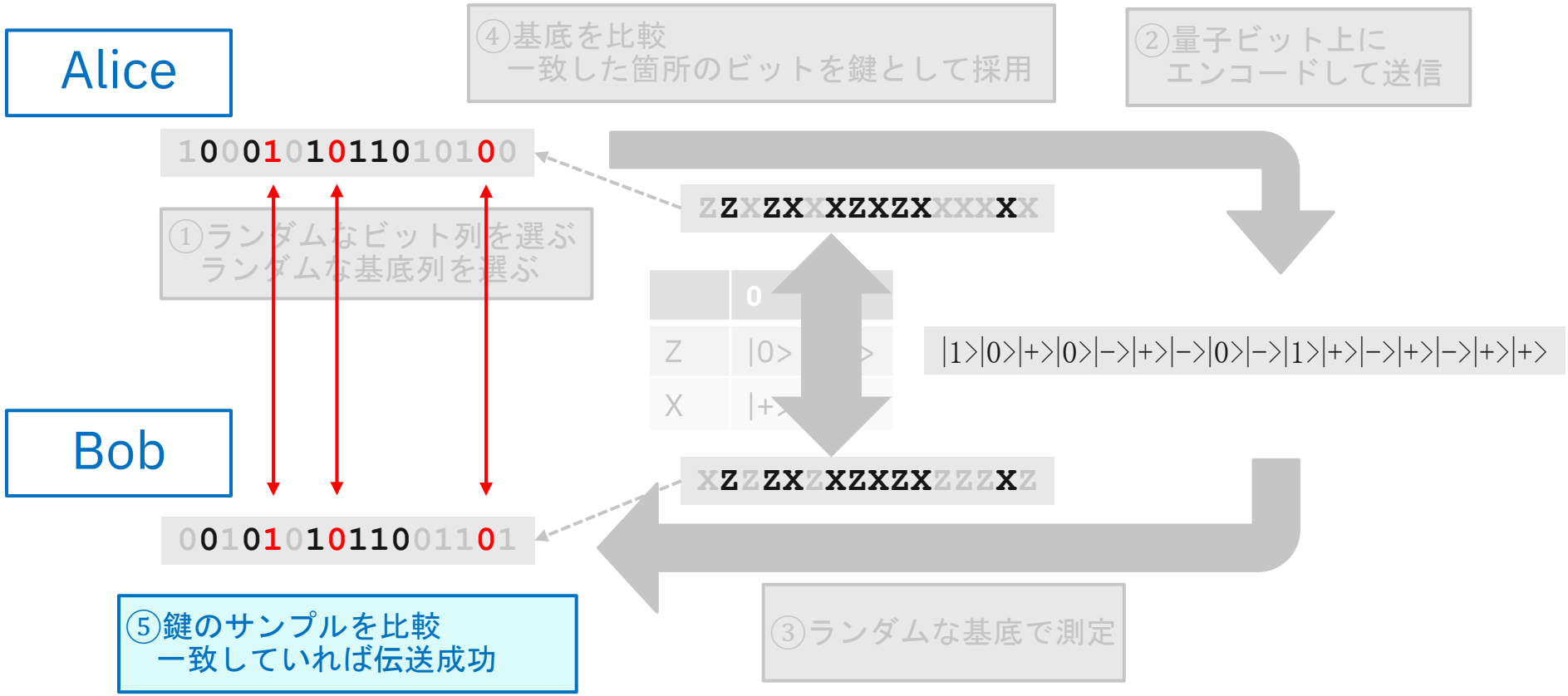
XZZZXZXZXXZZZXZ

③ランダムな基底で測定

2. プロトコルの概要


Alice
④基底を比較
一致した箇所のビットを鍵として採用
②量子ビット上に
エンコードして送信
10001010110101000
Z ZX XZXZX XXX X
①ランダムなビット列を選ぶ
ランダムな基底列を選ぶ
0
Z |0>
X |+>
|1>|0>|+>|0>|−>|+>|−>|0>|−>|1>|+>|−>|+>|−>|+>|+>
Bob
Z ZX XZXZX ZZZ Z
00101010110001101
③ランダムな基底で測定

2. プロトコルの概要


Alice
④基底を比較
一致した箇所のビットを鍵として採用
②量子ビット上に
エンコードして送信
1 0 0 0 1 0 1 0 1 1 0 1 0 1 0 0
Z ZX XZXZX XXX X
①ランダムなビット列を選ぶ
ランダムな基底列を選ぶ
0
Z |0>
X |+>
|1>|0>|+>|0>|−>|+>|−>|0>|−>|1>|+>|−>|+>|−>|+>|+>
Bob
Z ZX XZXZX ZZZ Z
0 0 1 0 1 0 1 0 1 1 0 0 1 1 0 1
⑤鍵のサンプルを比較
一致していれば伝送成功
③ランダムな基底で測定

3. Qiskitの例：盗聴なし


```
np.random.seed(seed=0)
n = 100

## Step 1
# Alice generates bits
alice_bits = randint(2, size=n)

## Step 2
# Create an array to tell us which qubits
# are encoded in which bases
alice_bases = randint(2, size=n)
message = encode_message(alice_bits, alice_bases)

## Step 3
# Decide which basis to measure in:
bob_bases = randint(2, size=n)
bob_results = measure_message(message, bob_bases)

## Step 4
alice_key = remove_garbage(alice_bases, bob_bases, alice_bits)
bob_key = remove_garbage(alice_bases, bob_bases, bob_results)

## Step 5
sample_size = 15
bit_selection = randint(n, size=sample_size)

bob_sample = sample_bits(bob_key, bit_selection)
print("  bob_sample = " + str(bob_sample))
alice_sample = sample_bits(alice_key, bit_selection)
print("alice_sample = "+ str(alice_sample))
```

3. Qiskitの例：盗聴なし


```
np.random.seed(seed=0)
n = 100

## Step 1
# Alice generates bits
alice_bits = randint(2, size=n)

## Step 2
# Create an array to tell us which qubits
# are encoded in which bases
alice_bases = randint(2, size=n)
message = encode_message(alice_bits, alice_bases)

## Step 3
# Decide which basis to measure in:
bob_bases = randint(2, size=n)
bob_results = measure_message(message, bob_bases)

## Step 4
alice_key = remove_garbage(alice_bases, bob_bases, alice_bits)
bob_key = remove_garbage(alice_bases, bob_bases, bob_results)

## Step 5
sample_size = 15
bit_selection = randint(n, size=sample_size)

bob_sample = sample_bits(bob_key, bit_selection)
print("  bob_sample = " + str(bob_sample))
alice_sample = sample_bits(alice_key, bit_selection)
print("alice_sample = "+ str(alice_sample))
```

Aliceが送る
メッセージ長

ランダムな
ビット列

ランダムな基底列で
エンコード

alice_bits

```
[0 1 1 0 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 0 1 1 0 0 1 1 1 1 0 1 0 1 0 1 1 0
 1 1 0 0 1 0 1 1 1 1 1 1 0 1 0 1 1 1 1 1 0 1 0 0 1 1 0 1 0 1 0 0 0 0 0 1 1 0 0
 0 1 1 0 1 0 0 1 0 1 1 1 1 1 1 1 0 1 1 0 0 1 0 0 1 1 0]
```

alice_bases

```
[1 0 0 1 0 0 0 1 1 0 1 0 0 0 0 0 1 0 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 0 0 1 0
 0 0 0 1 1 0 0 1 0 1 1 1 1 0 0 0 1 0 1 1 1 0 1 0 0 1 0 1 1 0 0 1 0 1 0 1 0
 1 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0]
```

3. Qiskitの例：盗聴なし

ビット列を基底列でエンコード

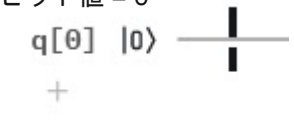
```
def encode_message(bits, bases):
    message = []
    for i in range(n):
        qc = QuantumCircuit(1,1)
        if bases[i] == 0: # Prepare qubit in Z-basis
            if bits[i] == 0:
                pass
            else:
                qc.x(0)
        else: # Prepare qubit in X-basis
            if bits[i] == 0:
                qc.h(0)
            else:
                qc.x(0)
                qc.h(0)
        qc.barrier()
        message.append(qc)
    return message
```

	0	1
Z	|0>	|1>
X	|+>	|->

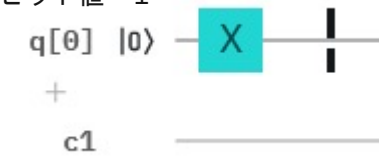
Z基底の場合

Quantum Tokyo

ビット値 = 0

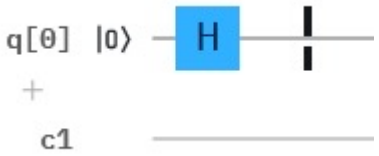


ビット値 = 1

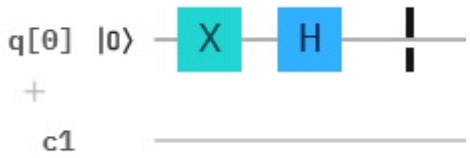


X基底の場合

ビット値 = 0



ビット値 = 1



3. Qiskitの例：盗聴なし


```
np.random.seed(seed=0)
n = 100

## Step 1
# Alice generates bits
alice_bits = randint(2, size=n)

## Step 2
# Create an array to tell us which qubits
# are encoded in which bases
alice_bases = randint(2, size=n)
message = encode_message(alice_bits, alice_bases)

## Step 3
# Decide which basis to measure in:
bob_bases = randint(2, size=n)
bob_results = measure_message(message, bob_bases)

## Step 4
alice_key = remove_garbage(alice_bases, bob_bases, alice_bits)
bob_key = remove_garbage(alice_bases, bob_bases, bob_results)

## Step 5
sample_size = 15
bit_selection = randint(n, size=sample_size)

bob_sample = sample_bits(bob_key, bit_selection)
print("  bob_sample = " + str(bob_sample))
alice_sample = sample_bits(alice_key, bit_selection)
print("alice_sample = "+ str(alice_sample))
```

Aliceが送る
メッセージ長

ランダムな
ビット列

ランダムな基底列で
エンコード

ランダムな基底列で
測定

alice_bits

```
[0 1 1 0 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 0 1 1 0 0 1 1 1 1 0 1 0 1 0 1 1 0
 1 1 0 0 1 0 1 1 1 1 1 1 0 1 0 1 1 1 1 1 0 1 0 0 1 1 0 1 0 1 0 0 0 0 0 1 1 0 0
 0 1 1 0 1 0 0 1 0 1 1 1 1 1 1 1 0 1 1 0 0 1 0 0 1 1 0]
```

alice_bases

```
[1 0 0 1 0 0 0 1 1 0 1 0 0 0 0 0 1 0 1 0 1 1 1 1 1 0 1 1 1 1 0 1 1 0 0 1 0
 0 0 0 1 1 0 0 1 0 1 1 1 1 0 0 0 1 0 1 1 1 0 1 0 0 1 0 1 1 0 0 1 0 1 0 1 0
 1 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0]
```

bob_bases

```
[1 0 1 0 0 1 1 0 0 0 1 1 0 0 0 0 0 1 0 1 0 0 0 1 1 1 0 0 1 1 1 1 0 0 0 1 1
 0 1 0 0 1 0 1 1 1 1 0 0 0 1 1 1 0 1 1 1 1 0 0 1 1 0 0 0 1 1 0 1 1 1 1 1 0
 0 0 1 0 1 0 1 1 0 0 0 1 0 0 1 1 1 1 0 1 0 0 0 0 1 1]
```

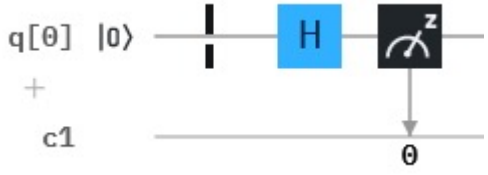
3. Qiskitの例：盗聴なし

メッセージを基底列で測定

```
def measure_message(message, bases):
    backend = Aer.get_backend('qasm_simulator')
    measurements = []
    for q in range(n):
        if bases[q] == 0: # measuring in Z-basis
            message[q].measure(0,0)
        if bases[q] == 1: # measuring in X-basis
            message[q].h(0)
            message[q].measure(0,0)
        result = execute(message[q], backend, shots=1, memory=True).result()
        measured_bit = int(result.get_memory()[0])
        measurements.append(measured_bit)
    return measurements
```

Z基底の場合

X基底の場合



3. Qiskitの例：盗聴なし


```
np.random.seed(seed=0)
n = 100

## Step 1
# Alice generates bits
alice_bits = randint(2, size=n)

## Step 2
# Create an array to tell us which qubits
# are encoded in which bases
alice_bases = randint(2, size=n)
message = encode_message(alice_bits, alice_bases)

## Step 3
# Decide which basis to measure in:
bob_bases = randint(2, size=n)
bob_results = measure_message(message, bob_bases)

## Step 4
alice_key = remove_garbage(alice_bases, bob_bases, alice_bits)
bob_key = remove_garbage(alice_bases, bob_bases, bob_results)

## Step 5
sample_size = 15
bit_selection = randint(n, size=sample_size)

bob_sample = sample_bits(bob_key, bit_selection)
print("  bob_sample = " + str(bob_sample))
alice_sample = sample_bits(alice_key, bit_selection)
print("alice_sample = "+ str(alice_sample))
```

Aliceが送る
メッセージ長

ランダムな
ビット列

ランダムな基底列で
エンコード

ランダムな基底列で
測定

基底列を比較
一致位置のビットを採用

alice_bits

```
[0 1 1 0 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 0 1 1 0 0 1 1 1 1 0 1 0 1 0 1 1 0
 1 1 0 0 1 0 1 1 1 1 1 1 0 1 0 1 1 1 1 1 0 1 0 0 1 1 0 1 0 1 0 0 0 0 0 1 1 0 0
 0 1 1 0 1 0 0 1 0 1 1 1 1 1 1 1 0 1 1 0 0 1 0 0 1 1 0]
```

alice_bases

```
[1 0 0 1 0 0 0 1 1 0 1 0 0 0 0 0 1 0 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 0 0 1 0
 0 0 0 1 1 0 0 1 0 1 1 1 1 1 0 0 0 1 0 1 1 1 0 1 0 0 1 0 1 1 0 0 1 0 1 0 1 0
 1 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0]
```

bob_bases

```
[1 0 1 0 0 1 1 0 0 0 1 1 0 0 0 0 0 1 0 1 0 0 0 1 1 1 0 0 1 1 1 1 1 0 0 0 1 1
 0 1 0 0 1 0 1 1 1 1 1 0 0 0 1 1 1 0 1 1 1 1 0 0 1 1 0 0 0 1 1 0 1 1 1 1 1 0
 0 0 1 0 1 0 1 1 0 0 0 1 0 0 1 1 1 1 0 1 0 0 0 0 1 1]
```

3. Qiskitの例：盗聴なし

基底列を比較して、一致しているところのビットのみ保持する

```
def remove_garbage(a_bases, b_bases, bits):
    good_bits = []
    for q in range(n):
        if a_bases[q] == b_bases[q]:
            # If both used the same basis, add
            # this to the list of 'good' bits
            good_bits.append(bits[q])
    return good_bits
```

3. Qiskitの例：盗聴なし


```
np.random.seed(seed=0)
n = 100

## Step 1
# Alice generates bits
alice_bits = randint(2, size=n)

## Step 2
# Create an array to tell us which qubits
# are encoded in which bases
alice_bases = randint(2, size=n)
message = encode_message(alice_bits, alice_bases)

## Step 3
# Decide which basis to measure in:
bob_bases = randint(2, size=n)
bob_results = measure_message(message, bob_bases)

## Step 4
alice_key = remove_garbage(alice_bases, bob_bases, alice_bits)
bob_key = remove_garbage(alice_bases, bob_bases, bob_results)

## Step 5
sample_size = 15
bit_selection = randint(n, size=sample_size)

bob_sample = sample_bits(bob_key, bit_selection)
print("  bob_sample = " + str(bob_sample))
alice_sample = sample_bits(alice_key, bit_selection)
print("alice_sample = "+ str(alice_sample))
```

Aliceが送る
メッセージ長

ランダムな
ビット列

ランダムな基底列で
エンコード

ランダムな基底列で
測定

基底列を比較
一致位置のビットを採用

サンプルビットを比較

alice_bits

```
[0 1 1 0 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 0 1 1 0 0 1 1 1 1 0 1 0 1 0 1 1 0
 1 1 0 0 1 0 1 1 1 1 1 1 0 1 0 1 1 1 1 0 1 0 0 1 1 0 1 0 1 0 0 0 0 0 1 1 0 0
 0 1 1 0 1 0 0 1 0 1 1 1 1 1 1 1 0 1 1 0 0 1 0 0 1 1 0]
```

alice_bases

```
[1 0 0 1 0 0 0 1 1 0 1 0 0 0 0 0 1 0 1 0 1 1 1 1 1 0 1 1 1 1 0 1 1 0 0 1 0
 0 0 0 1 1 0 0 1 0 1 1 1 1 0 0 0 1 0 1 1 1 0 1 0 0 1 0 1 1 0 0 1 0 1 0 1 0
 1 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0]
```

bob_bases

```
[1 0 1 0 0 1 1 0 0 0 1 1 0 0 0 0 0 1 0 1 0 0 0 1 1 1 0 0 1 1 1 1 0 0 0 1 1
 0 1 0 0 1 0 1 1 1 1 0 0 0 1 1 1 0 1 1 1 1 0 0 1 1 0 0 0 1 1 0 1 1 1 1 1 0
 0 0 1 0 1 0 1 1 0 0 0 1 0 0 1 1 1 1 0 1 0 0 0 0 1 1]
```

3. Qiskitの例：盗聴なし

ビット列からサンプルを取り出す

```
def sample_bits(bits, selection):
    sample = []
    for i in selection:
        # use np.mod to make sure the
        # bit we sample is always in
        # the list range
        i = np.mod(i, len(bits))
        # pop(i) removes the element of the
        # list at index 'i'
        sample.append(bits.pop(i))
    return sample
```

指定位置のビットを取り出してサンプルにする
元のビット列からは除去

3. Qiskitの例：盗聴なし


```
np.random.seed(seed=0)
n = 100

## Step 1
# Alice generates bits
alice_bits = randint(2, size=n)

## Step 2
# Create an array to tell us which qubits
# are encoded in which bases
alice_bases = randint(2, size=n)
message = encode_message(alice_bits, alice_bases)

## Step 3
# Decide which basis to measure in:
bob_bases = randint(2, size=n)
bob_results = measure_message(message, bob_bases)

## Step 4
alice_key = remove_garbage(alice_bases, bob_bases, alice_bits)
bob_key = remove_garbage(alice_bases, bob_bases, bob_results)

## Step 5
sample_size = 15
bit_selection = randint(n, size=sample_size)

bob_sample = sample_bits(bob_key, bit_selection)
print("  bob_sample = " + str(bob_sample))
alice_sample = sample_bits(alice_key, bit_selection)
print("alice_sample = "+ str(alice_sample))
```

Aliceが送る
メッセージ長

ランダムな
ビット列

ランダムな基底列で
エンコード

ランダムな基底列で
測定

基底列を比較
一致位置のビットを採用

サンプルビットを比較

```
  bob_sample = [0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
alice_sample = [0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
```

一致

alice_bits

```
[0 1 1 0 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 0 1 1 0 0 1 1 1 1 0 1 0 1 0 1 1 0
 1 1 0 0 1 0 1 1 1 1 1 1 0 1 0 1 1 1 1 1 0 1 0 1 0 0 0 0 0 1 1 0 0
 0 1 1 0 1 0 0 1 0 1 1 1 1 1 1 1 0 1 1 0 0 1 0 0 1 1 0]
```

alice_bases

```
[1 0 0 1 0 0 0 1 1 0 1 0 0 0 0 0 1 0 1 0 1 1 1 1 1 0 1 1 1 1 0 1 1 0 0 1 0
 0 0 0 1 1 0 0 1 0 1 1 1 1 0 0 0 1 0 1 1 1 0 1 0 0 1 0 1 1 0 0 1 0 1 0 1 0
 1 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0]
```

bob_bases

```
[1 0 1 0 0 1 1 0 0 0 1 1 0 0 0 0 0 1 0 1 0 0 0 1 1 1 0 0 1 1 1 1 0 0 0 1 1
 0 1 0 0 1 0 1 1 1 1 0 0 0 1 1 1 0 1 1 1 1 0 0 1 1 0 0 0 1 1 0 1 1 1 1 1 0
 0 0 1 0 1 0 1 1 0 0 0 1 0 0 1 1 1 1 0 1 0 0 0 0 1 1]
```

```
bob_sample == alice_sample
```

```
True
```

```
print(bob_key)
print(alice_key)
print("key length = %i" % len(alice_key))
```

```
[1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0,
[1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0,
key length = 33
```

鍵長

4. Qiskitの例：盗聴あり

```
np.random.seed(seed=3)
## Step 1
alice_bits = randint(2, size=n)
## Step 2
alice_bases = randint(2, size=n)
message = encode_message(alice_bits, alice_bases)
## Interception!!
eve_bases = randint(2, size=n)
intercepted_message = measure_message(message, eve_bases)
## Step 3
bob_bases = randint(2, size=n)
bob_results = measure_message(message, bob_bases)
## Step 4
bob_key = remove_garbage(alice_bases, bob_bases, bob_results)
alice_key = remove_garbage(alice_bases, bob_bases, alice_bits)
## Step 5
sample_size = 15
bit_selection = randint(n, size=sample_size)
bob_sample = sample_bits(bob_key, bit_selection)
print("  bob_sample = " + str(bob_sample))
alice_sample = sample_bits(alice_key, bit_selection)
print("alice_sample = "+ str(alice_sample))
```

盗聴

4. Qiskitの例：盗聴あり

```
np.random.seed(seed=3)
## Step 1
alice_bits = randint(2, size=n)
## Step 2
alice_bases = randint(2, size=n)
message = encode_message(alice_bits, alice_bases)
## Interception!!
eve_bases = randint(2, size=n)
intercepted_message = measure_message(message, eve_bases)
## Step 3
bob_bases = randint(2, size=n)
bob_results = measure_message(message, bob_bases)
## Step 4
bob_key = remove_garbage(alice_bases, bob_bases, bob_results)
alice_key = remove_garbage(alice_bases, bob_bases, alice_bits)
## Step 5
sample_size = 15
bit_selection = randint(n, size=sample_size)
bob_sample = sample_bits(bob_key, bit_selection)
print("  bob_sample = " + str(bob_sample))
alice_sample = sample_bits(alice_key, bit_selection)
print("alice_sample = "+ str(alice_sample))
```

```
  bob_sample = [1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0]
alice_sample = [1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
```

```
bob_sample == alice_sample
```

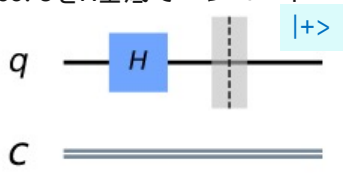
```
False
```

一致しない → 盗聴されていると判断

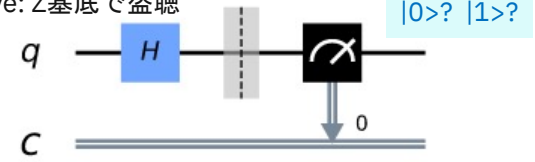
盗聴

Messageの状態（例）

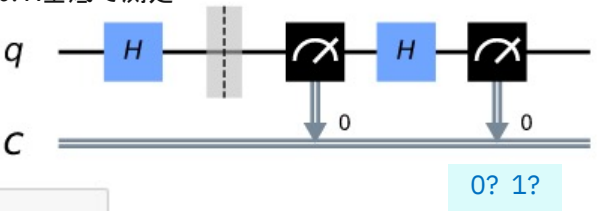
Alice: **0**をX基底でエンコード



Eve: Z基底で盗聴



Bob: X基底で測定

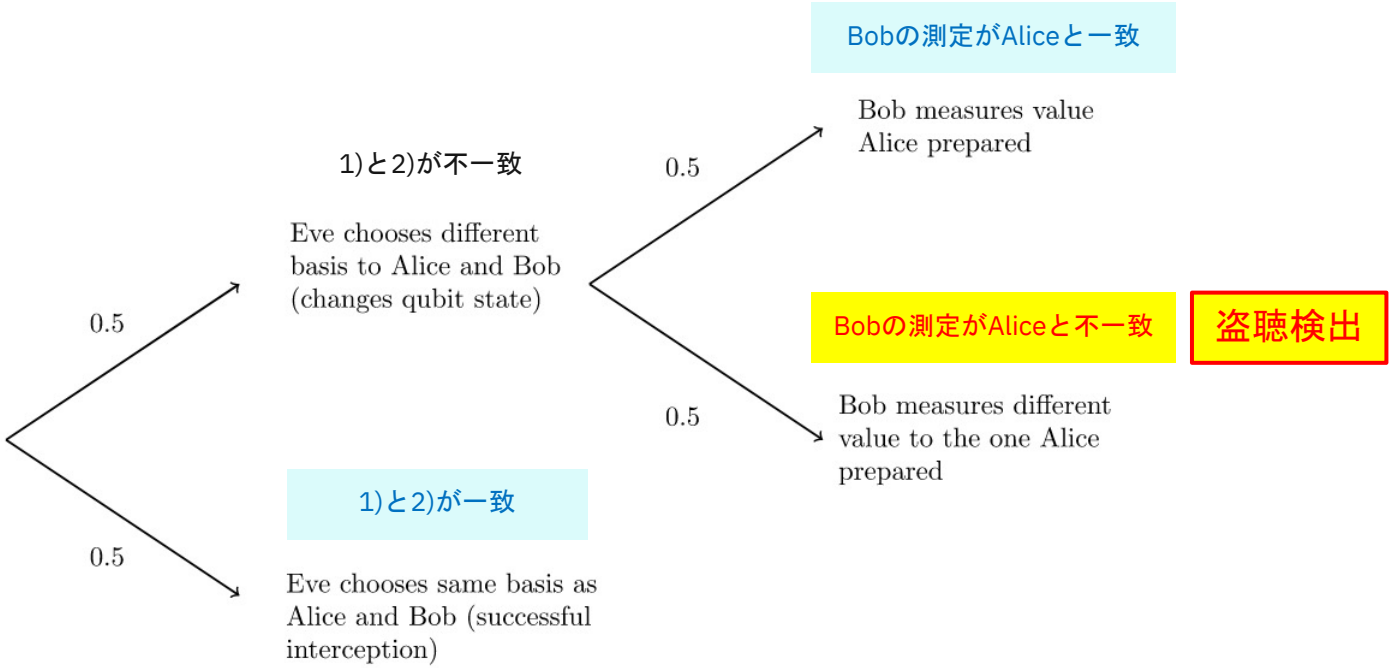


5. リスク分析

1) AliceとBobが選んだ基底

vs

2) Eveが選んだ基底



盗聴が気づかれない確率 $P(\text{undetected}) = 0.75^x$

15ビット比較 → 1.3%

50ビット比較 → 0.00006%

Thank you

Shun Shirakawa
Application Architect

wakarasi@jp.ibm.com

© Copyright IBM Corporation 2020. All rights reserved. The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. Any statement of direction represents IBM's current intent, is subject to change or withdrawal, and represent only goals and objectives. IBM, the IBM logo, and ibm.com are trademarks of IBM Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available at Copyright and trademark information.