Integrating Quantum Concepts into Cybersecurity

Activity 2: Qiskit (Quantum Information Science Kit)

Qiskit Programs – Joseph Spring

The following tutorial is based on Abraham Asfaw's Teleportation tutorial on Qiskit and YouTube

Exercise

1. Draw the standard Teleportation Circuit

Tutorial

1. The following code is a variation on the previous examples of code that you have seen for including libraries. Type in the code and select Run. Compare the code we used previously for this stage



2. The following code defines the type of quantum circuit that we wish to use for the teleportation protocol

🖸 Quan	S Jose; S Jose; S Sign 1 S Willie P Pure S Corei S www. Hom. S Sign 1 S Welc 💆 Quar A IBM 6 Q IB X S Quar Upht +
$\leftarrow \rightarrow$	C 🕯 quantum-computing.ibm.com/jupyter/user/Untitled2.ipynb 🔍 🖈 b 🔤 \varTheta 🗄
≡	IBM Q Experience
ඛ	Trusted Kernel 🔿 💭 Jupyter
	File Edit View Insert Cell Kernel Widgets Help
≣	B + ※ 绍 L ↑ ↓ ► Run ■ C ➡ Code ▼ □ ○ ペ
타	In [1]: 1 from qiskit import *
().	In [2]: 1 # We start by defining a circuit with three qubits and three classical bits
	<pre>2 circuit = QuantumLircuit(3, 3) 3 # The following code enables us to draw the circuit</pre>
?	<pre>4 %matplotlb inline 5 circuit.draw[output = 'mpl']]</pre>
(

This has the following output for the circuit. Three qubits q_0 , q_1 , q_2 and c representing the three classical bits, c_0 , c_1 , and c_2 : Note that each qubit commences with the zero qubit | 0 >

👂 🖸 Qu	ian 🔕 Josep	🕲 Josep 🔕	Sign I 🔕 Willia	∂ Pure Ø C	orei 🛛 🕙 www.	Hom: 🔇	Sign 🗌 🚱 Wel	c 🔯 Quan	🚓 IBM 🤇	Q IE ×	😂 Qiski 🔛	There Light	+		×
\leftarrow	C 🗎 o	quantum-compi	uting.ibm.com/jupy	rter/user/Untitled	2.ipynb							Q	☆	b 🔤 🛛 😆) :
≡	IBM Q Expe	rience	♡ 1_start_l	nere.ipy	③ Getting Starte	ed	♡ Untitled2	ipynb x						٩	Î
습	File	Edit Vie	ew Insert	Cell Kern	el Widgets	Help					Trusted	Kernel 🔵	\bigcirc	jupyter	r
		+ × 4	⊾ ↑ ↓	► Run 🔳	C 🏕 Cod	le	¥	• *							
≣		In [2]:	1 # We s	tart by defi	ning a circui	t with tl	hree qubits	and three	classica	al bits					
맠			2 circui 3 # The	<pre>t = QuantumC following co otlib inline</pre>	ircuit(3, 3) de enables us	to draw	the circui	t							
Ĭ)			5 circui	t.draw(outpu	t = 'mpl')										
?			q_0	—											
			q_1	—											1
			q ₂	_											
			С	²											
	_														
		1	R 🕈				22	202	202	1	F	os 🖂 🚺 🔊 🛠	Pr all	(*) (** 21:0 27/11/2	2 2019

The next stage in the code deals with the defining an example of an unknown state. We want to start with the unknown qubit state | 1 >, so we need to apply the Pauli NOT gate X to q₀:

🗖 Qu	uan 🛛 🔕 Josep	🕲 Josep 🔕	Sign I 🛛 🕙 Willia	Pure S Corei	S www.	🚱 Sign :	🕙 Welc: 🛛 📴 Quar	i 📥 IBM 🤇 💽 IE 🗙	😂 Qiskii 🛛 🔛	There Light	+			×
$\leftarrow \rightarrow$) C 🔒	quantum-compu	ting.ibm.com/jupy	ter/user/Untitled2.ipyn	b					Q	☆	b tais	Θ	:
≡	IBM Q Expe	erience	♡ 1_start_h	ere.ipy 💿 G	etting Started	🕽 Unt	itled2.ipynb x						ĉ	Î
습	File	Edit View	w Insert	Cell Kernel	Widgets Help	р			Trusted	Kernel 🔵	\bigcirc	jupy	/ter	I
		+ * 4	₿ ↑ ↓	► Run 🔳 C	➢ Code	•	× 0 🕬							
		In [3]:	1 # This	stage deals wit	h setting up ar	n example	of an unknown	n quantum state						
박			3 # Here	we insert a bar	rier between th	he first	two stages of	the circuit						
Ċ).			5 # And	now the code to	draw the circui	it at the	current stage	e is included						
					office Di									d
0			q_0 q_1	- x										
			<i>q</i> ₂	3										I
			С											ľ
		- 111			1(1)									-
1			R 🕈		۲ 🗃		Sec. 2. 1	March	- -	a 🖂 🚺 🗊 🗱 I	e al e	» 🛱	21:21 27/11/201	9

4. The next stage is to define the EPR (Bell) state that we will use in the teleportation protocol. This is defined between the sender (q_1) and receiver (q_2) . The circuit shown is the output for this stage of the protocol

🗾 Qua	an 🔕 Josep	0	Josep	🕄 Sig	n i 🕄	Willia	Pure -	🕄 Cor	eF 🔕	www.	Home	🕄 S	ign i 🛛 🕄	Welc	📴 Qu	an 🖂 🕹 i	IBM (QIE	×	🕒 Qis	kit 🛛 🛄	There	Light	+) ×
\leftrightarrow \rightarrow	C 🔒	quant	tum-co	mputin	g.ibm.c	:om/jupy	ter/user/l	Intitled2.	ipynb														Q	. ☆	٥	talls	e :
≡	IBM Q Exp	erieno	ce		Ç 1	_start_h	iere.ipy	. @) Gett	ing Star	ted	0	Untitl	ed2.i	oynb x												Â
습	File	Ed	lit	View	In	sert	Cell	Kernel	W	/idgets	Help	D								Т	rusted	Ke	rnel 🔿	0) آز	ipyt	er
	B	+	ж	@	G	↑ ↓	▶ Ru	n 🔳	c)	Co	ode		▼ 🖽	3 (o *												
≣			In	[4]:	1	# Defi	ning th	e EPR	(Bell) stat	e. Firs	st we	apply	the	Hadama	ard gat	te ar	nd the	en t	he CN	'OT ga	te					
₽					2 3	circui circui	t.h(1) t.cx(1,	2)																			- 1
0					4 5	# And circui	now the t.draw(<i>code</i> output	<i>to dra</i> = 'm	aw the pl')	circui	it at	the c	urrei	nt sta _l	(e is i	inclu	ıded									- 1
				1																							- 1
0						q ₀ q ₁	- <u>x</u>		H	-																	l
						q 2				-0-																	-1
						С	3				_																
																											_
()		0: \	P		R	-	X	0		F) 1					80		8	2		T (] 🖨 🕻		<mark>™ ≣</mark> *	Pr at	1 🕪	Ê ₽ 27/.	11:36 11/2019

5. The following code defines the first part of the Teleportation Protocol

🖸 Qu	ian 🔕 Josep	ol 🕲	isep 🛛 🕄 S	Sign I 🔕 Wi	lia 🗦 Pure	S Core	3 www.	Home	Sigr	i 🛛 🕄 We	lc 🛛 💆 Qua	in 🛛 📣 IBM	Q IE >	< 🖨 (Qiskit 🛛 🏪	There	Light	+			x
\leftarrow) C 🔒	quantur	m-comput	ing.ibm.com/	jupyter/user/l	Untitled2.ipy	/nb										Q	☆	b 🔤	0	:
≡	IBM Q Exp	erience		Ç 1_sta	rt_here.ipy.	💿	Getting S	itarted	Ω.	Untitled2	.ipynb x									ዖ	^
ώ	File	Edit	Viev	v Insert	Cell	Kernel	Widge	ts Help	p						Trusted	Kern	el 🔿	\bigcirc	jup	yter	
	B	+	፦ የነ	r 🖒	↓ Ru	in 🔳 C	* ••	Code	•		0 *										
≣			In [5]:	1 # H 2 cir	ere we ind cuit.barri	<i>lude a b</i> .er()	arrier	between 1	the EPK	? circui	t and th	e standa	rd Telep	oortat.	ion Ciı	cuit					
막				3 # 7 4 cir	he first p cuit.cx(0,	part of t 1)	he Tele	portation	n Proto	ocol											
Ŭ).				5 cir 6 # A 7 cir	cuit.h(0) <i>nd now the</i> cuit.draw <mark>(</mark>	e <i>code to</i> loutput =	draw t	the circu	it at t	the curr	ent stag	e is inc	luded								
0					70 - x	_	н —			н											ł
					12	_															4
					$c \rightarrow \frac{3}{2}$																ľ
		0: <u>\</u>	P	R 1		0	F			22	20		2.1	Î.	0 5	05 🖂 🚺	a * 1	∎⊭ .atl	4) ∰	21:46 27/11/20	019

6. The next stage is to measure each of the qubits at the sender's side (at q_0 and q_1).



Question. What does the 0 and 1 denote on the classical part of this circuit?

From this point on explain what is happening for each line of code.

7. Additional gates are added to the circuit; a CNOT gate and a Controlled Z gate



Question. What are the additional gates doing?

8. Finally we add the following code



Question. What is each line of the above code doing?

9. The above code outputs the following histogram:



Question. What does the histogram show?

Exercise

How would the above code change if the unknown state was the | +> state rather than a | 1 > state?

Circuit Composer

The circuit composer is an option available to help develop skills in creating quantum circuits in the IBM Q Experience.

In this section I would like you to create a new circuit. This should give you access to two tutorials. The first being the Composer Guide and the second a Gates overview. The composer guide runs through the *Hello Quantum World* Program. This we have met as the EPR circuit.

- 1. For your first circuit I would like you to Create the EPR circuit. Save and run the circuit. You will be asked for the backend that you would like to use. Does it matter which you use?
- 2. Now work through the Composer Guide. Compare the approach taken in the composer guide to the approach that you took with the EPR circuit above.
- 3. Now work through the Gates Overview. Which gates are familiar and which are not so familiar?

References

- 1. <u>https://quantum-computing.ibm.com/support/guides/getting-started-with-qiskit</u> <u>notebooks?section=5cd027ee555a22003bf5957e#</u>
- 2. <u>https://www.youtube.com/watch?v=mMwovHK2NrE</u>