

**Secure Programming**

**Simple Hash Chain**

**Primrose Nyagwaya (B0099882)**

***School of Informatics***

***Department of Informatics and Engineering***

***Institute of Technology, Blanchardstown***

***Dublin 15.***

**Actual Word Count: 777**

**Bachelor of Science**

**Computing in Information Technology**

**19/04/2019**

Contents

[Introduction 2](#_Toc6653458)

[Scenario: 2](#_Toc6653459)

[Procedure: 2](#_Toc6653460)

[Answers 3](#_Toc6653461)

[Joodle.com 3](#_Toc6653462)

[Hashes in python 4](#_Toc6653463)

[Finding ‘c89aa2ffb9edcc6604005196b5f0e0e4’ 5](#_Toc6653464)

[Conclusion 7](#_Toc6653465)

[References 8](#_Toc6653466)

[Figure 1Printing hello world on joodle 5](#_Toc6653491)

[Figure 2 hello world hashing 6](#_Toc6653492)

[Figure 3Calculating hashes 7](#_Toc6653493)

[Figure 4‘c89aa2ffb9edcc6604005196b5f0e0e4’ found 7](#_Toc6653494)

[Figure 5 Seed hash 8](#_Toc6653495)

# Introduction

For this lab a python code is used to solve a simple hash chain or blockchain problem. Hash function accepts a series of bytes and returns 128 bit hash value. One of the important things about the hash is that it should be difficult for someone to figure out what the input data was for a given hash. In a hash chain you simply start with a number then you hash it then you hash the result and then hash that result. You end up getting a series of hash values. So normally a hash chain is a hash value that we hash again, and again to produce a chain.

# Scenario:

You've registered for an online service that uses hash chains.

You've registered as user 'nOOB’ and have been given the hash chain seed 654e1c2ac6312d8c6441282f155c8ce9

Use the given information to figure out how to authenticate as the user 'ECSC' for the given challenge hash c89aa2ffb9edcc6604005196b5f0e0e4 i.e. Find the hash that hashes to this - This hash will be your solution.

# Procedure:

1. Get your python environment set up and run a hello world sample to get yourself started. Goto https://www.jdoodle.com/ and type the following code and execute it. print "Hello World!"
2. Next we want to see how to calculate hashes in python. You should be able to use the following bit of code to calculate the MD5 of the the string “Hello World!” import hashlib some\_string = "Hello World!" hash = hashlib.md5() hash.update(some\_string) print (hash.hexdigest())
3. Our next step is to understand what a hash chain is. So normally a hash chain is a hash value that we hash again, and again etc to produce a chain A = MD5(‘seed’) B = MD5(A) C = MD5(B) Etc. Or MD5(MD5(MD5(‘seed’)) This produces a chain of hashes, and we keep going until the hash that we get is equal to the hash we are looking for (The challenge hash), so in our example we need to find ‘c89aa2ffb9edcc6604005196b5f0e0e4’
4. The final piece of the jigsaw is to understand the ‘seed’ value. The seed value is the initial starting point of the hash chain. It’s the original string that we hash and might normally be a user’s password or similar. For our example this last step has a bit of a trick in it.. I suggest googling the seed given for the user nOOB and try to figure out how the seed value for this user is generated. Once you figure this out you need to calculate the seed for the user ECSC and then work out the hash chain until you reach the challenge hash.
5. The solution is the hash that actually maps to the challenge hash, so the second last hash in our chain as such.

# Answers

## Joodle.com

The first part of the lab was to go on joodle.com and print hello world. Below is a screenshot of execution of print world. The hello world program ran successfully, and we see the Hello World printed. (python3-programming-online, n.d.)

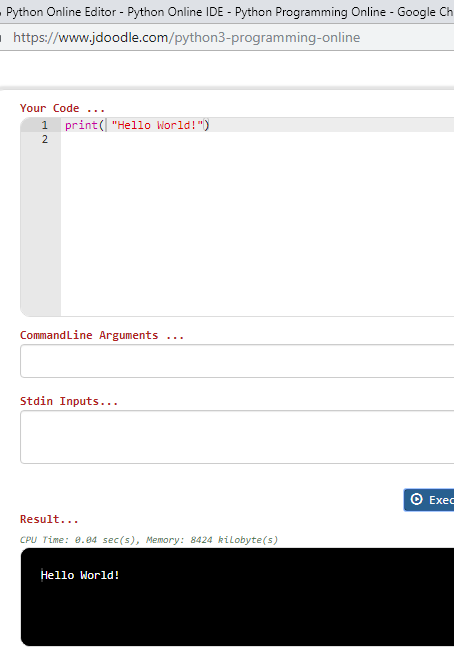


Figure 1Printing hello world on joodle

## Hashes in python

The second part of the lab is to calculate hashes in python. First thing we must import the hashlib library. Below is a code to calculate hashes in python. The given string to be calculated is Hello World. After executing the code below, we see the hashing of hello world.

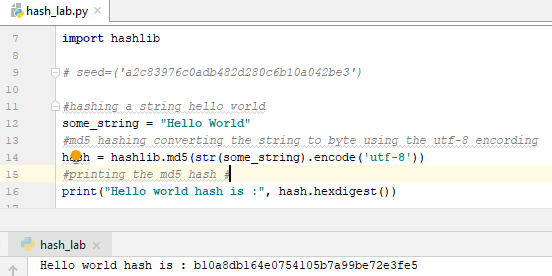


Figure 2 hello world hashing

## Finding ‘c89aa2ffb9edcc6604005196b5f0e0e4’

For calculating this we were provided with some details

* hash chain seed 654e1c2ac6312d8c6441282f155c8ce9
* challenge hash c89aa2ffb9edcc6604005196b5f0e0e4

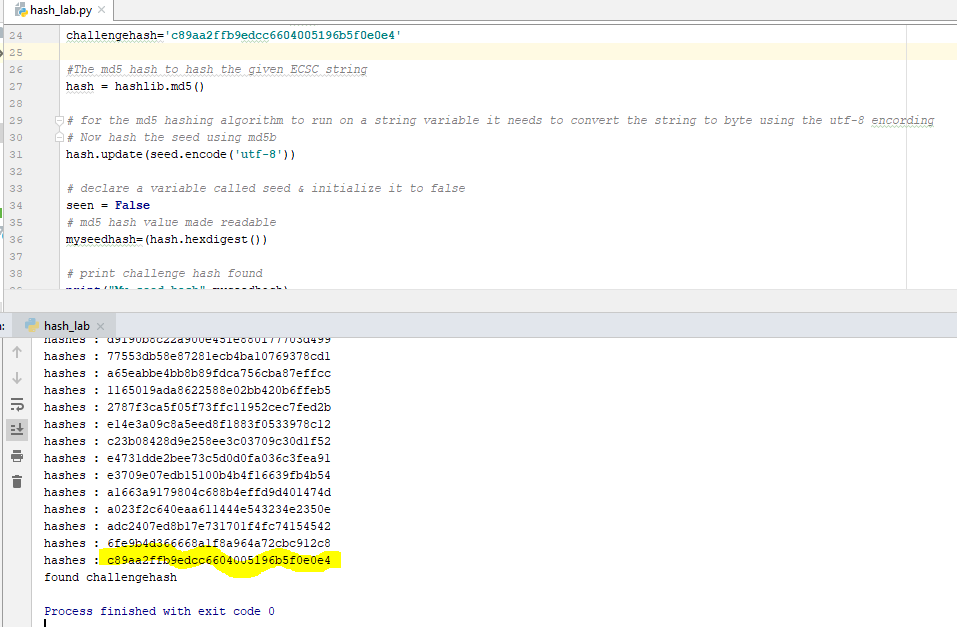
Functions

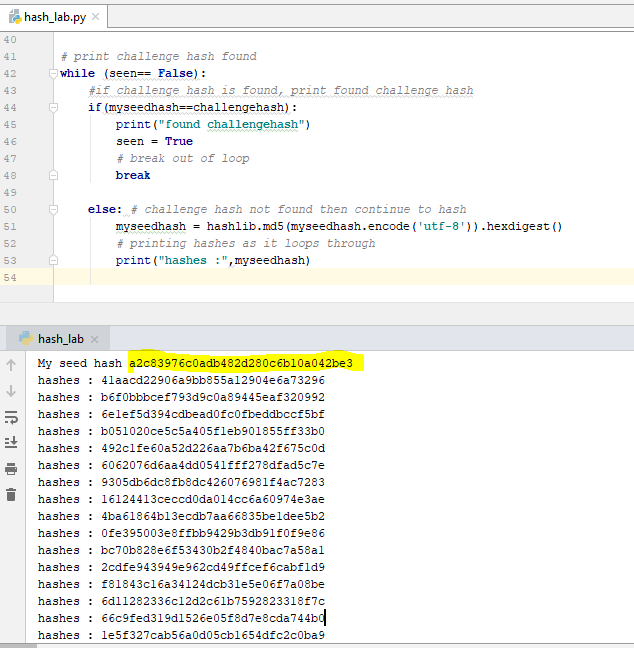
* encode() : Converts the string into bytes to be acceptable by hash function.
* digest() : Returns the encoded data in byte format.
* hexdigest() : Returns the encoded data in hexadecimal format.



Figure 3Calculating hashes

After executing the above code we look for ‘c89aa2ffb9edcc6604005196b5f0e0e4’

Figure 4‘c89aa2ffb9edcc6604005196b5f0e0e4’ found

Figure 5 Seed hash

# Conclusion

Calculating hashes is a very interesting topic to work on. Getting exposed of how things work in the background is an eye opener for me. Putting it together was difficult but all in all a good experience

# References

*hashlib.html*. (n.d.). Retrieved from docs.python.org: https://docs.python.org/2/library/hashlib.html

*md5-hash-python*. (n.d.). Retrieved from www.geeksforgeeks.or: https://www.geeksforgeeks.org/md5-hash-python/

*python3-programming-online*. (n.d.). Retrieved from www.jdoodle.com: https://www.jdoodle.com/python3-programming-online

*unicode*. (n.d.). Retrieved from docs.python.org: https://docs.python.org/3/howto/unicode.html