Fintech Hw7

b04902028 CSIE grade 4 Hao Hsiang, Hung

January 2019

1 Evaluate 4G

x = 10338857399563508035974916425421659830878883530402360147780309523428649499368 3
y = 37057141145242123013015316630864329550140216928701153669873286428255828810018

2 Evaluate 5G

x = 21505829891763648114329055987619236494102133314575206970830385799158076338148
y = 98003708678762621233683240503080860129026887322874138805529884920309963580118

3 Evaluate Q= dG

x = 19292789413885822208795898529305149840182528658437072379106007877942422663179
y = 85523802914842987620789296500568519676390717070621931282164943219424747867739

4 With standard Double-and-Add algorithm for scalar multiplications, how many doubles and additions respectively are required to evaluate dG?

(1). d base 10 = 902028

(2). d base 2 = 11011100001110001100

(3). First 1 is initialized, so 19 doubles and 9 adds are required.

5 Note that it is effortless to find -P from any P on a curve. If the addition of an inverse point is allowed, try your best to evaluate dG as fast as possible. Hint: 31P= 2(2(2(2(2P)))) -P

Scan from left to right. If the next bit is 1, then do the addition first, and multiply the same time as the number of consequence 1. When get 0 bit after a consequential 1, then add -P. If meet 0 bit, then just do multiplication. Take d for example, first bit is 1, so initialize 1 and get 1. There are 2 1's before 0, so we can multiply twice and get 100 and minus 1 to get 11. We get 0 at third bit, so just multiply once and get 110. Next bit is 1, so add 1 first and repeat the process again. We can get 110+1 = 111 -¿ multiply to get 111000 -¿ minus to get 111000-1 -¿ 110111. In the end, we can get dG after repeat the porcess. This method can reduce the time cost of adding because the whole times of adding of original algorithm should be larger or equal to this method. The new result is: double 19 times and add 7 times.

6 Take a Bitcoin transaction as you wish. Sign the transaction with a random number k and your private key d

(1). k = 7122

(2). transaction = 1206

(3). z = 69713921940860486323236596947059886945947039811974664044868154789621553632297

(4). kG =
(32668507330458402105453509073726043941996130509523329438135215467552629386872,
28410341433600605794812220093131039199158884047930401859847399207210945513034)

(5). r = x1 mod n
= 32668507330458402105453509073726043941996130509523329438135215467552629386872

(6). s = $k^{-1} * (z + r * d)$ mod n
= 72502973898521024559599012446596003432322735751578821033294785240582050764453

7 Verify the digital signature with your public key Q

(1). w = s^{-1} mod n
= 10651044001294999191723171754329095539095695023611110432424336574320817816797 4

(2). u1 = zw mod n
= 768449534489462613040362901777693120131522573346493770203739623252708677653 06

(3). u2 = rw mod n
= 198295324489329781408339934029102536861230642588236116121169651745605157256 4

(4). u1 * G + u2 * Q =
(326685073304584021054535090737260439419961305095233294381352154675526293868 72
,
284103414336006057948122200931310391991588840479304018598473992072109455130 34
)

(5). r = 326685073304584021054535090737260439419961305095233294381352154675526293868 72
x1 = 326685073304584021054535090737260439419961305095233294381352154675526293868 72
r = x1 mod n