Comparing the Speeds of HMAC and CMAC

資管三 B01705009 巫奕萱 資管三 B01705041 郝晉凱

I. Implementation

In the proposal, we've already mentioned in brief how we will implement. This time, we will describe it in more detail. Our purpose is to test the speed of 7 different MAC generation ways: HMAC with SHA-224, SHA-256, SHA-384, SHA-512 and CMAC with AES-128, AES-192, AES-256. Besides, the program is written in Python, using online open source references.

Each time we randomly generate keys for each method. The key length for HMAC is selectable. And the key length for CMAC depends on the kinds of AES, that is to say, the key size is 128 bits for AES-128, 192 bits for AES-192, etc. After the keys are selected, we start a timer and record the end time for each algorithm. Then we know how much time each process takes. For the case of fixed message size, the message size is 2048 bits. The process will repeated 100 times and generate a table at the same time.

Each time we randomly generate keys for each method. The key length for HMAC is selectable. And the key length for CMAC depends on the kinds of AES, that is to say, the key size is 128 bits for AES-128, 192 bits for AES-192, etc. After the keys are selected, we start a timer and record the end time for each algorithm. Then we know how much time each process takes. The program will generate a table. Each row indicates the different plain text size from 2^{13} to 2^{20} bits with distance of 2^{13}bits.

Both methods use the time as the denominator. Then we can compare the speed of HMAC and CMAC. The bigger value indicated that the method is faster.

II. Result and Implication—for fixed message size

Assume that the content of message will not influence the speed of HMAC and CMAC. Since we don't know whether different key size will have impacts on the result, we divided this case into four parts—HMAC & CMAC with AES-128, HMAC & CMAC with AES-192, HMAC & CMAC with AES-256 (key size is the same as the corresponding CMAC for the above cases), and the comparison between three different CMAC. For each case, the process will repeat 100 times.

For the first case, HMAC & CMAC with AES-128, the key lengths are all 128 bits. First, we have to check whether they are normally distributed. H_0: The population of the speed for method i is normally distributed; H_1: the population of the speed for method i is not normally distributed. α = 0.05, d.f. = 2.

Chi-Squared Test of Normality			
	sha-224(128)		
Mean	35608.85642		
Standard deviation	10195.6874		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	9
(-1.5 < z <= -0.5)	0.24173	24.173	19
(-0.5 < z <= 0.5)	0.382925	38.2925	35
(0.5 < z <= 1.5)	0.24173	24.173	36
(z > 1.5)	0.066807	6.6807	1
chi-squared Stat	12.8122		
df	2		
p-value	0.0017		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	sha-256(128)		
Mean	49010.82592		
Standard deviation	13038.7713		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	8
(-1.5 < z <= -0.5)	0.24173	24.173	19
(-0.5 < z <= 0.5)	0.382925	38.2925	28
(0.5 < z <= 1.5)	0.24173	24.173	45
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	28.7589		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	sha-384(128)		
Mean	48428.6803		
Standard deviation	14235.8165		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	11
(-1.5 < z <= -0.5)	0.24173	24.173	14
(-0.5 < z <= 0.5)	0.382925	38.2925	31
(0.5 < z <= 1.5)	0.24173	24.173	44
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	31.4056		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	sha-512(128)		
Mean	52083.46424		
Standard deviation	13749.5245		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	9
(-1.5 < z <= -0.5)	0.24173	24.173	16
(-0.5 < z <= 0.5)	0.382925	38.2925	40
(0.5 < z <= 1.5)	0.24173	24.173	35
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	15.1747		
df	2		
p-value	0.0005		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	cmac aes-128		
Mean	2808.599116		
Standard deviation	717.0382		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	10
(-1.5 < z <= -0.5)	0.24173	24.173	18
(-0.5 < z <= 0.5)	0.382925	38.2925	33
(0.5 < z <= 1.5)	0.24173	24.173	39
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	19.7322		
df	2		
p-value	0.0001		
chi-squared Critical	5.9915		

Since all 5 methods have p-value < 0.05 = α, we have overwhelming evidence to support the alternative hypothesis. Thus, according to chi-squared test for normality, we know none of these data are normal. Then we drew the histogram for each of them and found they are identical in shape and spread. It satisfies the required condition for Friedman Test. H_0: The locations of all 5 populations are the same; H_1: at least two populations differ; α = 0.05, d.f. = 4.

Friedman Test		
Group		Rank Sum
sha 224		210.5
sha 256		392
sha 384		349
sha 512		448.5
aes 128		100
Fr Stat		323.71
df		4
p-value		0
chi-squared Critical		9.4877

Since p-value < 0.05 = α, we have overwhelming evidence to support the alternative hypothesis. It means that the locations are not the same for all populations. Then we use Wilcoxon sign rank sum test to compare the population pair by pair. H_0: The two population locations are the same; H_1: The population1 is located to the left/right of population 2; α = 0.05

By comparing the z statistic and the z critical, we know the relationship between them.

Wilcoxon Signed Rank Sum Test		
Difference		*sha 224 - aes 128*
T+		5050
T-		
Observations (for test)		100
z Stat		8.682
P(Z<=z) one-tail		0
z Critical one-tail		1.6449
P(Z<=z) two-tail		0
z Critical two-tail		1.96

Wilcoxon Signed Rank Sum Test		
Difference		*sha 224 - sha 256*
T+		5
T-		4945
Observations (for test)		99
z Stat		-8.621
P(Z<=z) one-tail		0
z Critical one-tail		1.6449
P(Z<=z) two-tail		0
z Critical two-tail		1.96

Wilcoxon Signed Rank Sum Test		
Difference		*sha 256 - sha 384*
T+		1468
T-		1088
Observations (for test)		71
z Stat		1.089
P(Z<=z) one-tail		0.1381
z Critical one-tail		1.6449
P(Z<=z) two-tail		0.2762
z Critical two-tail		1.96

Wilcoxon Signed Rank Sum Test		
Difference		*sha 256 - sha 512*
T+		851
T-		2635
Observations (for test)		83
z Stat		-4.05
P(Z<=z) one-tail		0
z Critical one-tail		1.6449
P(Z<=z) two-tail		0
z Critical two-tail		1.96

Chi-Squared Test of Normality			
	sha-224(192)		
Mean	39698.34776		
Standard deviation	8237.7644		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	10
(-1.5 < z <= -0.5)	0.24173	24.173	9
(-0.5 < z <= 0.5)	0.382925	38.2925	46
(0.5 < z <= 1.5)	0.24173	24.173	33
(z > 1.5)	0.066807	6.6807	2
chi-squared Stat	19.2271		
df	2		
p-value	0.0001		
chi-squared Critical	5.9915		

Therefore, the speed of CMAC with AES-128 < HMAC with SHA-224 < HMAC with SHA-256 = HMAC with SHA-384 < HMAC with SHA-512.

For the second case, HMAC & CMAC with AES-192, the key lengths are all 192 bits. First, we have to check whether they are normally distributed. H_0: The population of the speed for method i is normally distributed; H_1: the population of the speed for method i is not normally distributed. α = 0.05, d.f. = 2.

Chi-Squared Test of Normality			
	sha-256(192)		
Mean	55035.26163		
Standard deviation	11385.3365		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	12
(-1.5 < z <= -0.5)	0.24173	24.173	5
(-0.5 < z <= 0.5)	0.382925	38.2925	51
(0.5 < z <= 1.5)	0.24173	24.173	32
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	32.8746		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	sha-384(192)		
Mean	53633.31917		
Standard deviation	11552.1185		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	13
(-1.5 < z <= -0.5)	0.24173	24.173	8
(-0.5 < z <= 0.5)	0.382925	38.2925	52
(0.5 < z <= 1.5)	0.24173	24.173	27
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	28.7162		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	sha-512(192)		
Mean	58031.75789		
Standard deviation	12270.9325		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	13
(-1.5 < z <= -0.5)	0.24173	24.173	4
(-0.5 < z <= 0.5)	0.382925	38.2925	53
(0.5 < z <= 1.5)	0.24173	24.173	30
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	36.5466		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	cmac aes-192		
Mean	2988.301952		
Standard deviation	624.4023		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	15
(-1.5 < z <= -0.5)	0.24173	24.173	9
(-0.5 < z <= 0.5)	0.382925	38.2925	45
(0.5 < z <= 1.5)	0.24173	24.173	31
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	29.6674		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Since all 5 methods have p-value < 0.05 = α, we have overwhelming evidence to support the alternative hypothesis. Thus, according to chi-squared test for normality, we know none of these data are normal.

Friedman Test		
Group		Rank Sum
sha 224		211
sha 256		374
sha 384		357
sha 512		458
aes 192		100
Fr Stat		326.44
df		4
p-value		0
chi-squared Critical		9.4877

Then we drew the histogram for each of them and found they are identical in shape and spread. It satisfies the required condition for Friedman Test.

H_0: The locations of all 5 populations are the same; H_1: at least two populations differ; α = 0.05, d.f. = 4.

Since p-value < 0.05 = α, we have overwhelming evidence to support the alternative hypothesis. It means that the locations are not the same for all populations. Then we use Wilcoxon sign rank sum test to compare the population pair by pair. H_0: The two population locations are the same; H_1: The population1 is located to the left/right of population 2; α = 0.05

By comparing the z statistic and the z critical, we know the relationship between them.

Wilcoxon Signed Rank Sum Test		
Difference		*aes 192 - sha 224*
T+		
T-		5050
Observations (for test)		100
z Stat		-8.682
P(Z<=z) one-tail		0
z Critical one-tail		1.6449
P(Z<=z) two-tail		0
z Critical two-tail		1.96

Wilcoxon Signed Rank Sum Test		
Difference		*sha 256 - sha 384*
T+		1728
T-		1047
Observations (for test)		74
z Stat		1.834
P(Z<=z) one-tail		0.0333
z Critical one-tail		1.6449
P(Z<=z) two-tail		0.0666
z Critical two-tail		1.96

Wilcoxon Signed Rank Sum Test		
Difference		*sha 224 - sha 384*
T+		85
T-		4965
Observations (for test)		100
z Stat		-8.39
P(Z<=z) one-tail		0
z Critical one-tail		1.6449
P(Z<=z) two-tail		0
z Critical two-tail		1.96

Wilcoxon Signed Rank Sum Test		
Difference		*sha 256 - sha 512*
T+		380.5
T-		2545.5
Observations (for test)		76
z Stat		-5.604
P(Z<=z) one-tail		0
z Critical one-tail		1.6449
P(Z<=z) two-tail		0
z Critical two-tail		1.96

Therefore, the speed of CMAC with AES-192 < HMAC with SHA-224 < HMAC with SHA-384 < HMAC with SHA-256 < HMAC with SHA-512.

For the third case, HMAC & CMAC with AES-256, the key lengths are all 256 bits. First, we have to check whether they are normally distributed. H_0: The population of the speed for method i is normally distributed; H_1: the population of the speed for method i is not normally distributed. α = 0.05, d.f. = 2.

Chi-Squared Test of Normality			
	sha-224（256)		
Mean	39709.65685		
Standard deviation	9191.3158		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	10
(-1.5 < z <= -0.5)	0.24173	24.173	20
(-0.5 < z <= 0.5)	0.382925	38.2925	39
(0.5 < z <= 1.5)	0.24173	24.173	30
(z > 1.5)	0.066807	6.6807	1
chi-squared Stat	8.6177		
df	2		
p-value	0.0134		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	sha-256(256)		
Mean	54105.10271		
Standard deviation	11151.3728		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	10
(-1.5 < z <= -0.5)	0.24173	24.173	15
(-0.5 < z <= 0.5)	0.382925	38.2925	50
(0.5 < z <= 1.5)	0.24173	24.173	22
(z > 1.5)	0.066807	6.6807	3
chi-squared Stat	10.9327		
df	2		
p-value	0.0042		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	sha-384(256)		
Mean	52087.44208		
Standard deviation	11174.6033		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	9
(-1.5 < z <= -0.5)	0.24173	24.173	16
(-0.5 < z <= 0.5)	0.382925	38.2925	45
(0.5 < z <= 1.5)	0.24173	24.173	29
(z > 1.5)	0.066807	6.6807	1
chi-squared Stat	10.5377		
df	2		
p-value	0.0051		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	sha-512(256)		
Mean	56394.5169		
Standard deviation	11389.7851		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	10
(-1.5 < z <= -0.5)	0.24173	24.173	20
(-0.5 < z <= 0.5)	0.382925	38.2925	31
(0.5 < z <= 1.5)	0.24173	24.173	39
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	19.5335		
df	2		
p-value	0.0001		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	cmac aes-256		
Mean	2968.018134		
Standard deviation	539.2198		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	6
(-1.5 < z <= -0.5)	0.24173	24.173	24
(-0.5 < z <= 0.5)	0.382925	38.2925	39
(0.5 < z <= 1.5)	0.24173	24.173	31
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	8.6925		
df	2		
p-value	0.013		
chi-squared Critical	5.9915		

Since all 5 methods have p-value < 0.05 = α, we have overwhelming evidence to support the alternative hypothesis. Thus, according to chi-squared test for normality, we know none of these data are normal.

Then we drew the histogram for each of them and found they are identical in shape and spread. It satisfies the required condition for Friedman Test.

Friedman Test		
Group		Rank Sum
sha 224		210
sha 256		388
sha 384		345.5
sha 512		456.5
aes 256		100
Fr Stat		329.626
df		4
p-value		0
chi-squared Critical		9.4877

H_0: The locations of all 5 populations are the same; H_1: at least two populations differ; α = 0.05, d.f. = 4. Since p-value < 0.05 = α, we have overwhelming evidence to support the alternative hypothesis. It means that the locations are not the same for all populations. Then we use Wilcoxon sign rank sum test to compare the population pair by pair. H_0: The two population locations are the same; H_1: The population1 is located to the left/right of population 2; α = 0.05. By comparing the z statistic and the z critical, we know the relationship between them.

Wilcoxon Signed Rank Sum Test		
Difference		*aes 256 - sha 224*
T+		
T-		5050
Observations (for test)		100
z Stat		-8.682
P(Z<=z) one-tail		0
z Critical one-tail		1.6449
P(Z<=z) two-tail		0
z Critical two-tail		1.96

Wilcoxon Signed Rank Sum Test		
Difference		*sha 224 - sha 384*
T+		182
T-		4868
Observations (for test)		100
z Stat		-8.056
P(Z<=z) one-tail		0
z Critical one-tail		1.6449
P(Z<=z) two-tail		0
z Critical two-tail		1.96

Wilcoxon Signed Rank Sum Test		
Difference		*sha 384 - sha 256*
T+		802.5
T-		1972.5
Observations (for test)		74
z Stat		-3.152
P(Z<=z) one-tail		0.0008
z Critical one-tail		1.6449
P(Z<=z) two-tail		0.0016
z Critical two-tail		1.96

Wilcoxon Signed Rank Sum Test		
Difference		*sha 256 - sha 512*
T+		779.5
T-		2541.5
Observations (for test)		81
z Stat		-4.148
P(Z<=z) one-tail		0
z Critical one-tail		1.6449
P(Z<=z) two-tail		0
z Critical two-tail		1.96

Therefore, the speed of CMAC with AES-256 < HMAC with SHA-224 < HMAC with SHA-384 < HMAC with SHA-256 < HMAC with SHA-512.

Chi-Squared Test of Normality			
	CMAC AES-128		
Mean	2895.061247		
Standard deviation	612.7248		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	14
(-1.5 < z <= -0.5)	0.24173	24.173	10
(-0.5 < z <= 0.5)	0.382925	38.2925	40
(0.5 < z <= 1.5)	0.24173	24.173	36
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	28.8722		
df	2		
p-value	0		
chi-squared Critical	5.9915		

For the last case, CMAC with AES-128, 192, and 256. First, we have to check whether they are normally distributed. H_0: The population of the speed for method i is normally distributed; H_1: the population of the speed for method i is not normally distributed. α = 0.05, d.f. = 2.

Chi-Squared Test of Normality			
	CMAC AES-192		
Mean	2943.853216		
Standard deviation	660.4299		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	16
(-1.5 < z <= -0.5)	0.24173	24.173	10
(-0.5 < z <= 0.5)	0.382925	38.2925	25
(0.5 < z <= 1.5)	0.24173	24.173	49
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	58.1035		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	CMAC AES-256		
Mean	2981.041731		
Standard deviation	650.7204		
Observations	100		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	6.6807	15
(-1.5 < z <= -0.5)	0.24173	24.173	10
(-0.5 < z <= 0.5)	0.382925	38.2925	31
(0.5 < z <= 1.5)	0.24173	24.173	44
(z > 1.5)	0.066807	6.6807	0
chi-squared Stat	43.0015		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Since all 3 methods have p-value < 0.05 = α, we have overwhelming evidence to support the alternative hypothesis. Thus, according to chi-squared test for normality, we know none of these data are normal.

Then we drew the histogram for each of them and found they are identical in shape and spread. It satisfies the required condition for Friedman Test.

Friedman Test		
Group		Rank Sum
CMAC AES-128		160
CMAC AES-192		208
CMAC AES-256		232
Fr Stat		26.88
df		2
p-value		0
chi-squared Critical		5.9915

H_0: The locations of all 3 populations are the same; H_1: at least two populations differ; $\alpha = 0.05$, d.f. = 2. Since p-value < 0.05 = α, we have overwhelming evidence to support the alternative hypothesis. It means that the locations are not the same for all populations. Then we use Wilcoxon sign rank sum test to compare the population pair by pair.

Wilcoxon Signed Rank Sum Test			
Difference	*CMAC AES-128 - CMAC AES-192*		
T+	1599		
T-	3252		
Observations (for test)	98		
z Stat	-2.929		
P(Z<=z) one-tail	0.0017		
z Critical one-tail	1.6449		
P(Z<=z) two-tail	0.0034		
z Critical two-tail	1.96		

Wilcoxon Signed Rank Sum Test			
Difference	*CMAC AES-192 - CMAC AES-256*		
T+	1829		
T-	2636		
Observations (for test)	94		
z Stat	-1.522		
P(Z<=z) one-tail	0.0641		
z Critical one-tail	1.6449		
P(Z<=z) two-tail	0.1282		
z Critical two-tail	1.96		

H_0: The two population locations are the same; H_1: The population1 is located to the left/right of population 2; $\alpha = 0.05$. By comparing the z statistic and the z critical, we know the relationship between them.

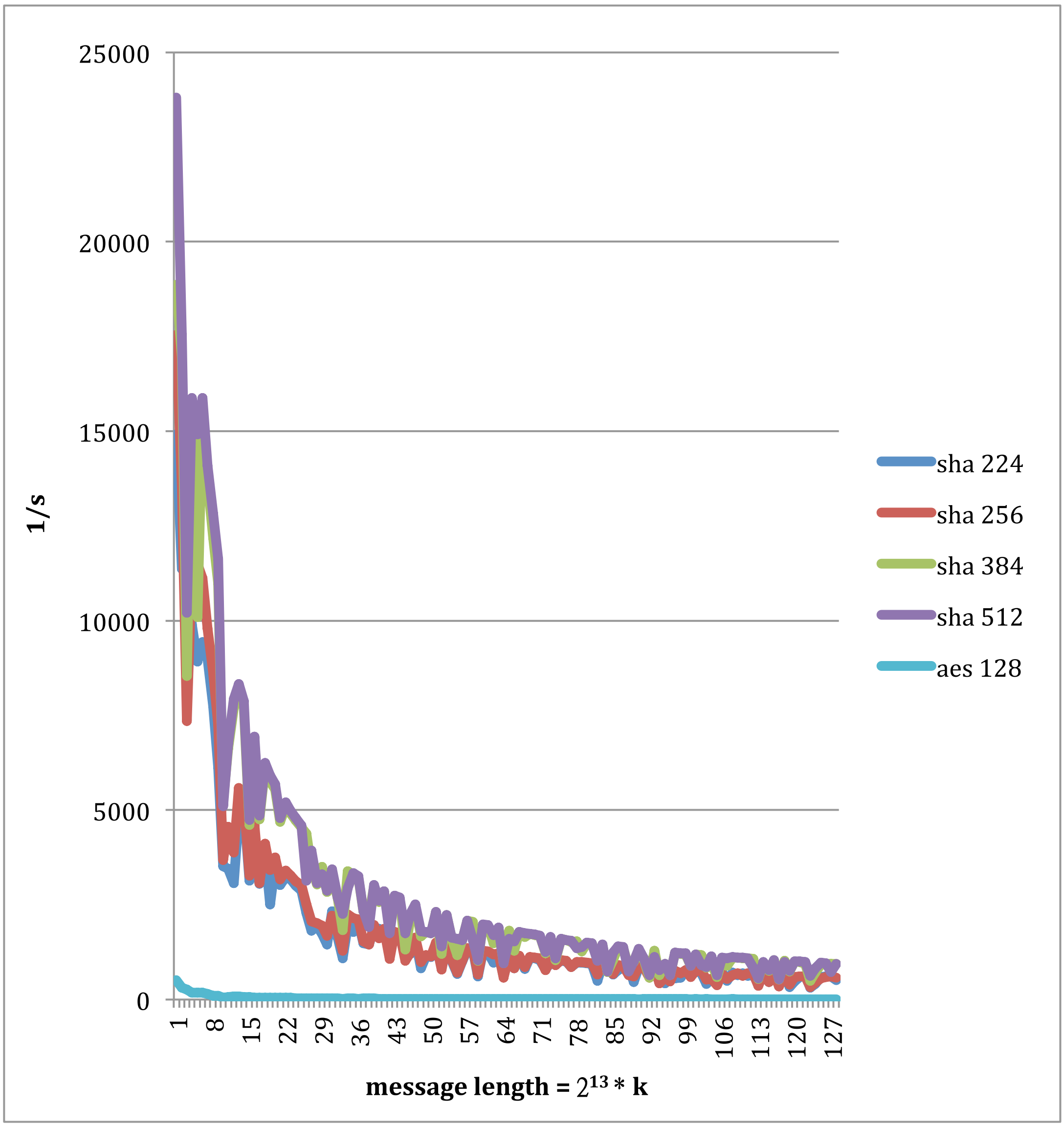
Therefore, the speed of CMAC with AES-128 < CMAC with AES-192 = CMAC with AES-256

According to the cases mentioned before, it is obvious that when the key lengths are the same, HMAC will always faster than CMAC. Also, among the four hash functions in SHA-2, SHA-512 is the fastest. Among the three cryptographic functions in AES, AES-256 is the fastest.

III. Result and implication –for increasing message size

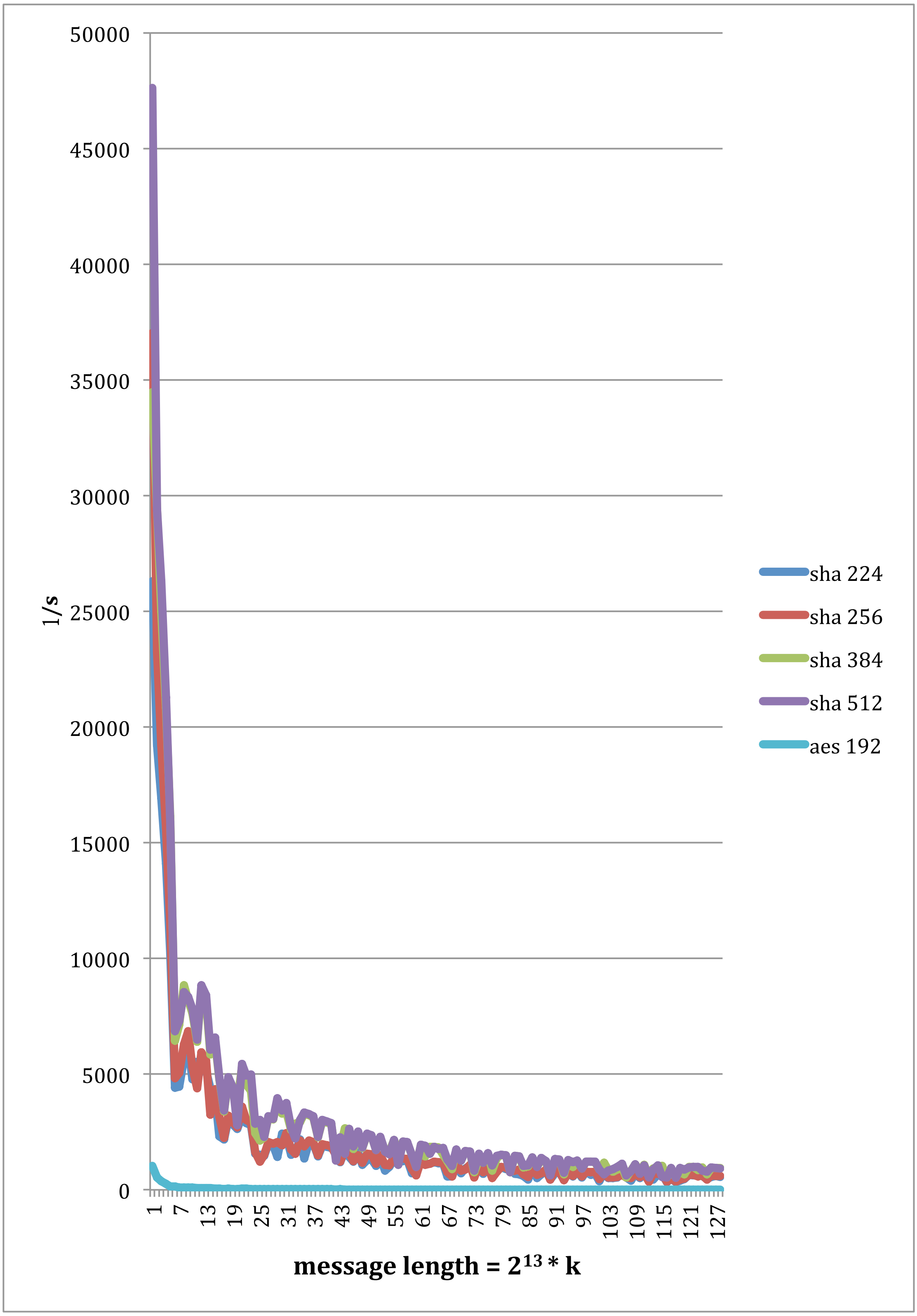
Assume that the content of message will not influence the speed of HMAC and CMAC. This time the variable is the length of message. We've already known that if the message size increases, the process will be more time consuming. This time, our purpose is to make sure the relationship we got from above will not alter owing to the change in length size. However, we still don't know whether different key size will have impacts on the result, we divided this case into four parts—HMAC & CMAC with AES-128, HMAC & CMAC with AES-192, HMAC & CMAC with AES-256 (key size is the same as the corresponding CMAC for the above cases), and the comparison between three different CMAC.

For the first case, HMAC & CMAC with AES-128, there is a line chart.



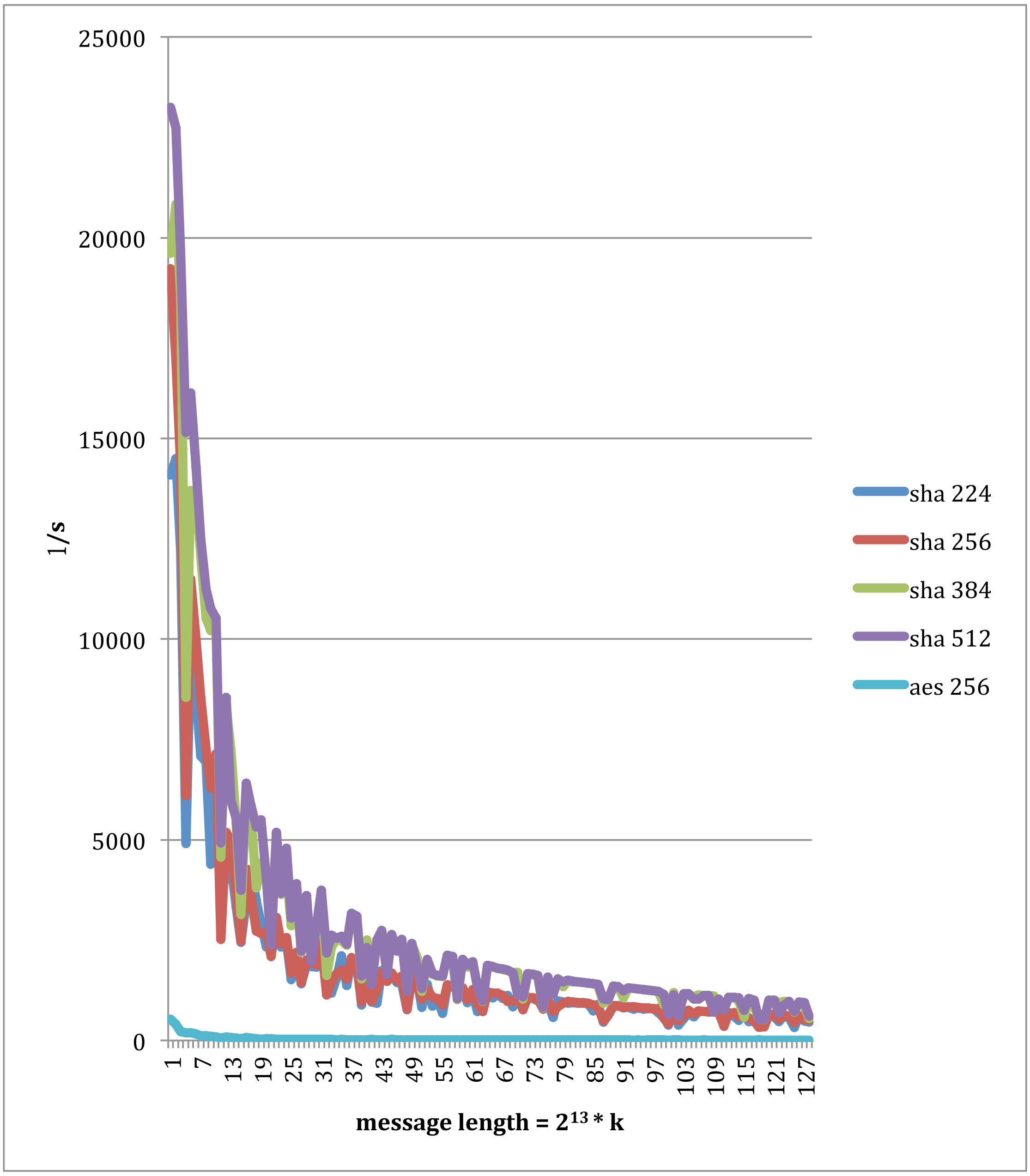
According to the diagram, CMAC is much slower than HMAC. Besides, SHA-512 is faster than SHA-224. But we cannot say the conclusion made on previous part "HMAC with SHA-224 < HMAC with SHA-256 = HMAC with SHA-384 < HMAC with SHA-512" for sure.

For the second case, HMAC & CMAC with AES-192, there is a line chart.



According to the diagram, CMAC is much slower than HMAC. Besides, SHA-512 is faster than SHA-224. But we cannot say the conclusion made on previous part "HMAC with SHA-224 < HMAC with SHA-384 < HMAC with SHA-256 < HMAC with SHA-512" for sure.

For the third case, HMAC & CMAC with AES-256, there is a line chart.



According to the diagram, CMAC is much slower than HMAC. Besides, SHA-512 is faster than SHA-224. But we cannot say the conclusion made on previous part "HMAC with SHA-224 < HMAC with SHA-384 < HMAC with SHA-256 < HMAC with SHA-512" for sure.

For the last case, CMAC with AES-128, 192, and 256, since the values are similar, we cannot get any conclusion from the line chart. Therefore, we still have to check them step by step.

First, we have to check whether they are normally distributed. H_0: The population of the speed for method i is normally distributed; H_1: the population of the speed for method i is not normally distributed. α = 0.05, d.f. = 2.

Chi-Squared Test of Normality			
	CMAC AES-128		
Mean	33.64054435		
Standard deviation	65.7139		
Observations	128		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	8.551296	0
(-1.5 < z <= -0.5)	0.24173	30.94144	0
(-0.5 < z <= 0.5)	0.382925	49.0144	116
(0.5 < z <= 1.5)	0.24173	30.94144	6
(z > 1.5)	0.066807	8.551296	6
chi-squared Stat	151.9048		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	CMAC AES-192		
Mean	35.4489673		
Standard deviation	71.6093		
Observations	128		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	8.551296	0
(-1.5 < z <= -0.5)	0.24173	30.94144	0
(-0.5 < z <= 0.5)	0.382925	49.0144	115
(0.5 < z <= 1.5)	0.24173	30.94144	7
(z > 1.5)	0.066807	8.551296	6
chi-squared Stat	147.6121		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Chi-Squared Test of Normality			
	CMAC AES-256		
Mean	35.0622036		
Standard deviation	71.4447		
Observations	128		
Intervals	Probability	Expected	Observed
(z <= -1.5)	0.066807	8.551296	0
(-1.5 < z <= -0.5)	0.24173	30.94144	0
(-0.5 < z <= 0.5)	0.382925	49.0144	118
(0.5 < z <= 1.5)	0.24173	30.94144	4
(z > 1.5)	0.066807	8.551296	6
chi-squared Stat	160.8066		
df	2		
p-value	0		
chi-squared Critical	5.9915		

Since all 3 methods have p-value < 0.05 = α, we have overwhelming evidence to support the alternative hypothesis. Thus, according to chi-squared test for normality, we know none of these data are normal.

Then we drew the histogram for each of them and found they are identical in shape and spread. It satisfies the required condition for Friedman Test. H_0: The locations of all 3 populations are the same;

H_1: at least two populations differ;

α = 0.05, d.f. = 2.

Since p-value = 0.9105 > α = 0.05, we have no sufficient evidence to support the alternative hypothesis. It means that the speeds of three methods are the same.

Friedman Test		
Group		Rank Sum
CMAC AES-128		260
CMAC AES-192		254
CMAC AES-256		254
Fr Stat		0.188
df		2
p-value		0.9105
chi-squared Critical		5.9915

According to the result got from these four cases, no matter how long the key size, the speeds of HMAC are fast than CMAC.

IV. Conclusion

Though the results from previous two parts have some difference, it is for sure that HMAC works faster than CMAC as the length of key are the same.

V. Appendix—Python Code

The Following is the Python code of the implementation of the method. For HMAC with SHA-224, SHA-256, SHA-384, SHA-512, the Python code is as follow:

```
def hmac_sha224(key, msg):
    start =  time.clock()
    hash_obj = hmac.new(key = key, msg = msg, digestmod = hashlib.sha224)
    hash_obj.hexdigest()
    return time.clock()-start

def hmac_sha256(key, msg):
    start =  time.clock()
    hash_obj = hmac.new(key = key, msg = msg, digestmod = hashlib.sha256)
    hash_obj.hexdigest()
    return time.clock()-start

def hmac_sha384(key, msg):
    start = time.clock()
    hash_obj = hmac.new(key = key, msg = msg, digestmod = hashlib.sha384)
    hash_obj.hexdigest()
    return time.clock()-start

def hmac_sha512(key, msg):
    start = time.clock()
    hash_obj = hmac.new(key = key, msg = msg, digestmod = hashlib.sha512)
    hash_obj.hexdigest()
    return time.clock()-start
```

While key refers to the key of the method, and msg is the sample message of the method. We use the standard library of Pyhton hmac and hashlib. For time calculating, we use Python time.clock() rather than time.time() which is more precise in UNIX system than the other. The above functions will return the time of generating the cipher code.

For CMAC, we use the Pyhton code as follow:

```
def cmac_aes(key, msg):
    start = time.clock()
    cipher = AES.new(key.decode('hex'), AES.MODE_CMAC)
    cipher.encrypt(msg).encode('hex')
    return time.clock()-start
```

Like HMAC, key and msg refers to key and messages used for CMAC-AES repectively, while the function return the time spent for the method as well. We use the 3rd-party library CryptoPlus.Cipher.AES for Python AES function.

For key Generating and message generating, We use the following code.

```
def keyGenerate(bits):
    return Random.get_random_bytes(bits/8).encode('hex')

def msgGenerate(bits):
    return Random.get_random_bytes(bits/8).encode('hex')
```

We use the library Crypto.Random for random number generating, the code is generated by bytes, so we divid the key size of 8.