Benchmarks

Verificatum JavaScript Cryptographic Library
Version: 1.1.1
Date: 2020-04-02
Browser: Firefox 1.0+ (best effort detection)

Each benchmark computes at least 3 samples of the measured quantity and takes the average. The running time of the layout engine is not included in the running time. Note that benchmarks give different results using different JavaScript engines and that other factors may influence the results. In particular, the last measurements may not reflect actual running time (it would be faster), since the garbage collection is poor on some platforms. Thus, interpret the results with care and investigate the particular case you are interested in before drawing any hard conclusions.

The library is designed with pre-computation in mind all the way up to the highest abstraction layer. Combined with a web worker as in this benchmark, such computations can be done in the background.

Exponentiation

The running times include the cost of generating random exponents, which gives an upper bound of the running time of the actual exponentiation.

The running time of modular exponentiation is increased by almost factor of 8 when the bit size of the modulus is doubled as expected from a relatively naive implementation. A similar behavior can be seen in the elliptic curves with growing field size, but with a slightly smaller factor.

Standard Multiplicative Groups

Group	ms / exp
modp768	22.3
modp1024	82.7
modp1536	50.3
modp2048	73.3
modp3072	390.7
modp4096	598.0
modp6144	1563.0
modp8192	3669.7

Standard Elliptic Curves

Group	ms / exp
prime192v1	26.0
prime192v2	12.3
prime192v3	7.3
prime256v1	13.3
prime239v1	10.0

prime239v3	9.7
secp192k1	11.7
secp192r1	8.3
secp224k1	20.0
secp224r1	8.0
secp256k1	13.3
secp256r1	11.3
secp384r1	17.7
secp521r1	32.3
brainpoolp192r1	19.0
brainpoolp224r1	6.7
brainpoolp256r1	10.0
brainpoolp320r1	15.0
brainpoolp384r1	21.7
brainpoolp512r1	39.0
P-192	4.3
P-224	5.3
P-256	8.0
P-384	16.0
P-521	31.0

Fixed-basis Exponentiation for Selected Groups

Here zero gives plain exponentiation for easy reference.

Group \ Exps	0	1	2	4	8	16	32
modp3072	237.7	715.0	175.8	111.9	79.7	61.6	48.6
modp4096	485.0	619.3	435.2	268.7	186.3	140.5	114.1
modp6144	1557.7	1944.7	1219.7	792.8	557.1	409.1	332.8

Encryption over Selected Groups

The running time of encryption grows linearly with the width, so the values for greater widths are readily extrapolated from the given numbers.

El Gamal Encryption (ms / ciphertext)

This is only benchmarked for the purpose of comparison. It is not CCA2 secure or even non-malleable, and should therefore not be used unless other equivalent mechanisms are in place.

Group \ Width	1	2	3	4
modp3072	397.3	803.3	1177.7	1547.3
modp4096	894.0	1773.0	2656.3	3544.3
modp6144	2787.3	5592.7	8474.3	11401.3
P-256	58.7	105.3	75.7	80.0
secp384r1	39.3	80.0	119.7	160.0

P-521	76.0	156.0	235.7	312.0

El Gamal Encryption with Label and ZKPoK (ms / ciphertext)

This is the simplest cryptosystem, which is non-malleable in a standard heuristic sense, i.e., it is the El Gamal cryptosystem with proof of knowledge of the randomness turned non-interactive using the Fiat-Shamir heuristic. This is not provably secure in the random oracle model, but likely to be secure.

Group \ Width	1	2	3	4
modp3072	565.0	1132.0	1745.0	2262.3
modp4096	1293.3	2581.3	3864.7	5153.7
modp6144	4072.7	8179.3	12226.7	16303.3
P-256	63.3	83.7	96.3	119.3
secp384r1	62.3	126.3	186.7	248.7
P-521	121.0	239.7	363.3	614.3

Naor-Yung with Label and ZKPoK (ms / ciphertext)

This is the Naor-Yung cryptosystem, which is provably CCA2 secure with the Fiat-Shamir heuristic in the random oracle model.

Group \ Width	1	2	3	4
modp3072	984.0	1970.7	2957.7	3942.3
modp4096	2256.0	4535.7	6779.7	9061.0
modp6144	7102.3	14200.0	21286.7	28469.0
P-256	72.3	146.0	214.3	289.0
secp384r1	167.0	334.7	503.0	672.0
P-521	353.0	704.7	1058.7	1409.0