Improvement of a JavaScript cryptographic library performance with big numbers

Master Semester Project

Julien von Felten

Supervisor: Gaylor Bosson
Responsible: Prof. Bryan Ford
DEDIS, EPFL

January 2020

Table of Contents

Introduction

Kyber

- Developed in Go
- Adapted in JS for status.dedis.ch
- JavaScript numbers < 2^{53}
- Uses bn.js library for big integers
- No verification of links for the SkipChain

Project

- Benchmark
- Optimizations
  - BigInt
  - Modulus
  - Pool

Each optimization implemented on top of each other

Benchmark

Library used: Performance MDN API
for browser performance

Measure of the time of:

- N signatures and each signature
- N verifications and each verification
  N = [2, 10, 100, 500, 1000]
- Min, Max, Avg computed

First optimization: BigInt

- BN.js: BNType, represented by array
- Replacing BNType by BigInt
- Implementation of more complex function than operators

Second optimization: Modulus

- 55% of time spent for modulus computation
- $(A \bmod n * B \bmod n) \bmod n = (A * B) \bmod n$
- Modulus functions used in higher abstraction level functions

Third optimization: Memory Pool

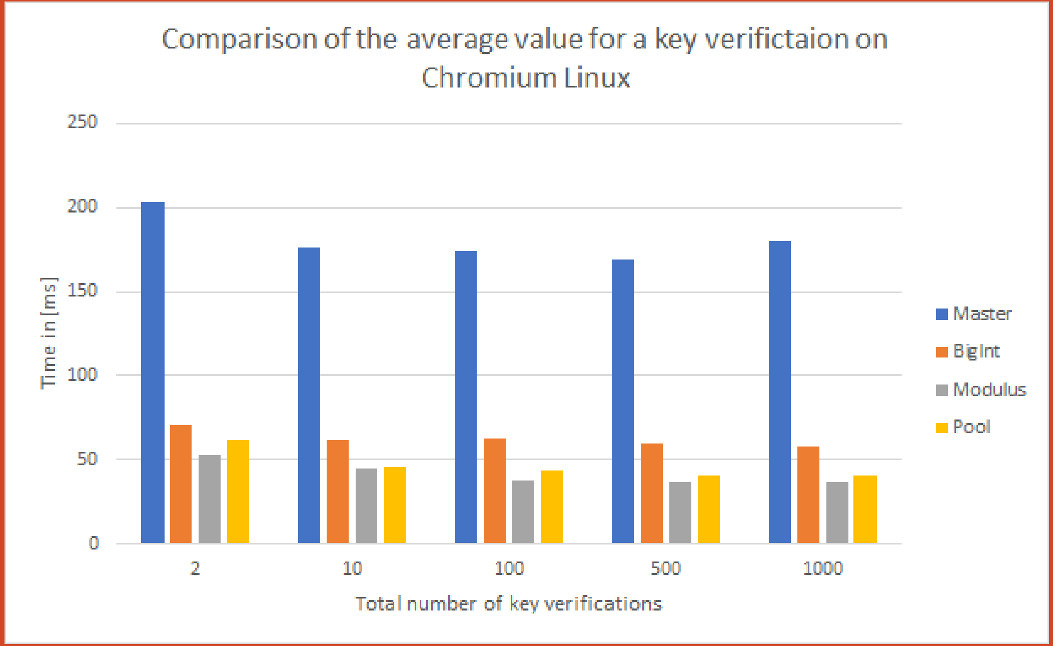
- Memory pool: Group of objects in memory ready to be used
- 318,000,000 gfp and 90,000,000 gfp2 objects created for N = 1000
- Package deePool and mutable classes

Results of the optimizations

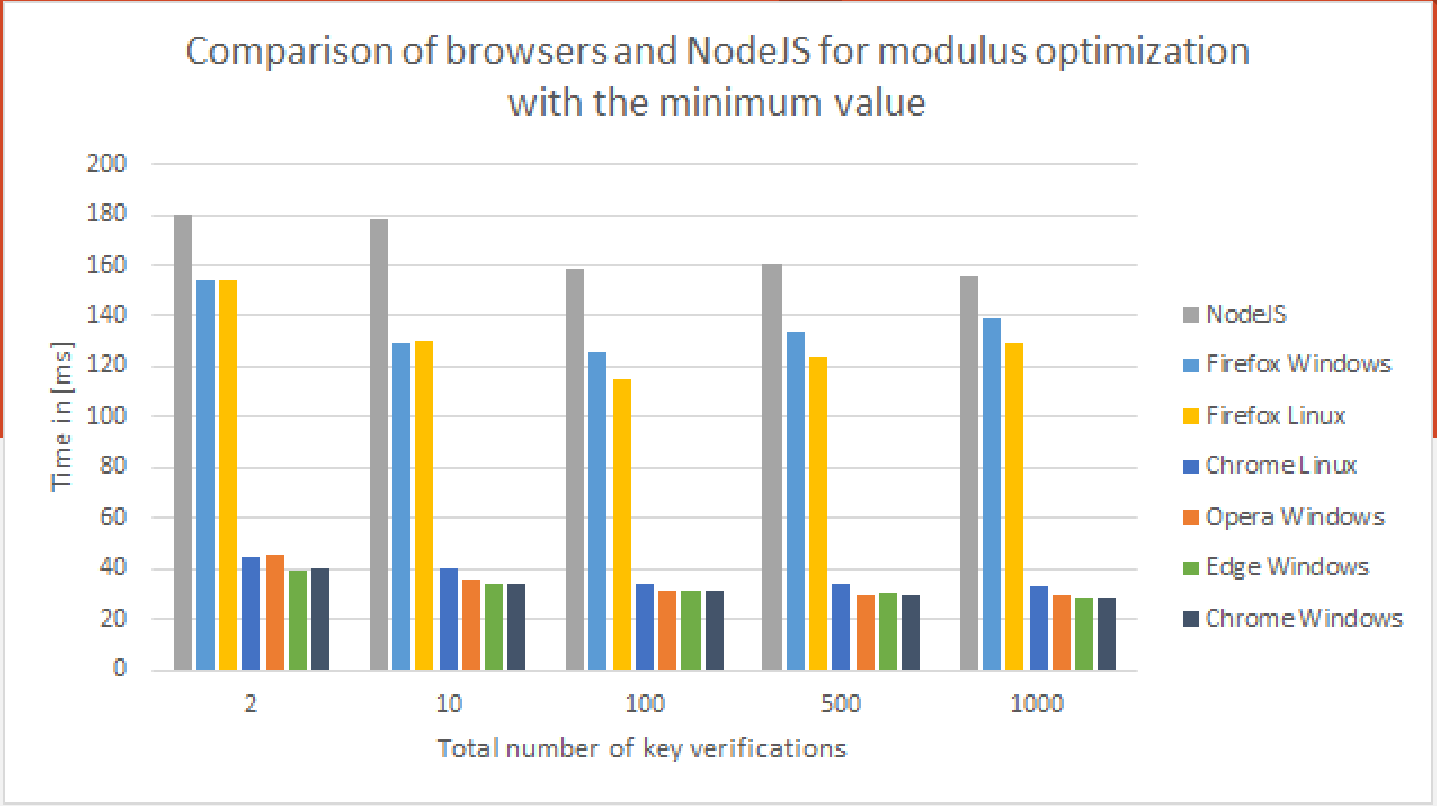
	Master	BigInt	Modulus	Pool
Minimum	166.3 [ms]	54.91 [ms]	33.36 [ms]	36.46 [ms]
Average	180.26 [ms]	58.21 [ms]	36.92 [ms]	41.08 [ms]
Maximum	222.79 [ms]	73.12 [ms]	55.9 [ms]	59.26 [ms]
Ratio	5.5	5.08	3.21	3.6

Ratio: time verification / time signature

Values obtained for 1000 verification keys



Results of the comparison of the browsers and NodeJS



- Chrome, Opera, Edge, NodeJs: v8 engine
- Firefox: SpiderMonkey engine
- NodeJS: more layers with event loop, low-level I/O API, file system I/O
- Best value obtained: 28.6ms on Chrome Windows

Future work

- Instead of BigInt: concatenation of two numbers
- Use even fewer modulus
- Different package of memory pool or own implementation
- Optimization for NodeJS

Conclusion

- BigInt type is working well but depends on browsers
- Fewer Modulus can be used
- Memory in JS is optimized, memory pool not needed
- Kyber can be improved more than 4.76
- Key verification under 30ms