Multi Agent Cognitive Architecture for Secure Computation in Internet of Things

Gajendra Deshpande, KLS Gogte Institute of Technology, Belagavi
Dr. Shrirang Kulkarni, National Institute of Engineering, Mysuru

Introduction

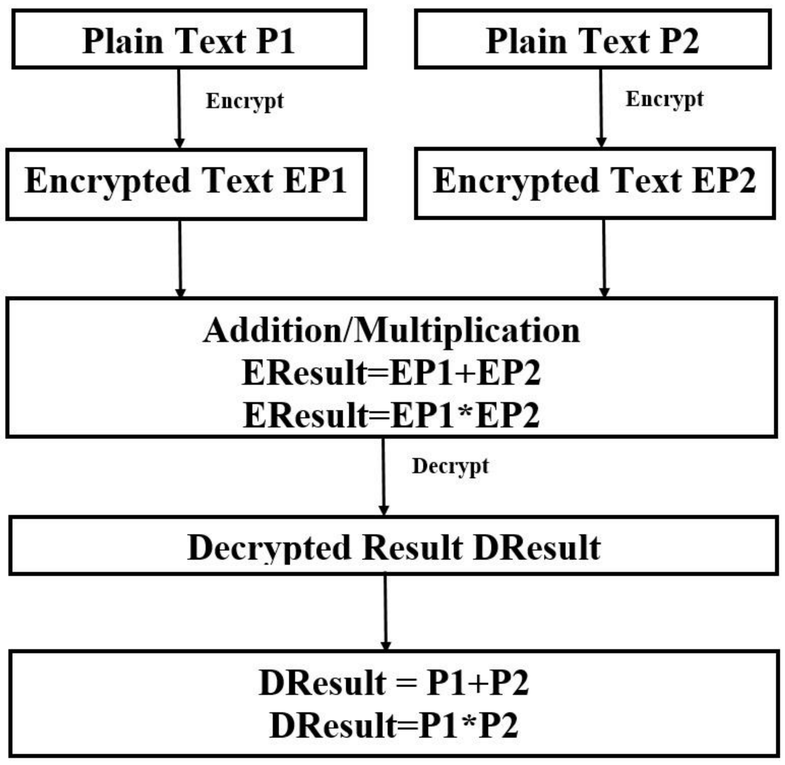
- Several billion devices are currently connected to the Internet, and this number will continue to grow.
- This is a consequence of not only more people becoming interested in consumer electronics but also more sensors and actuators being incorporated into everyday electronics, household appliances, and the general infrastructure.
- Since most of these devices are not able to process data locally, they will often upload it to a third party for processing.
- However, this data may be private, the third party may not be trustworthy, or both. Therefore, the data should be encrypted before it is transferred

- Imagine taking all of your credit card statements and locking them into a safe, to which you have the only key. Your statements are now protected from prying eyes. This is what encryption does.
- But what if you wanted to analyse your expenditure on groceries in the last 12 months? First you would have to unlock the safe and retrieve the statements. So now the documents are out in the open and they can be read by anyone. This is what decryption does.
- The difference with Homomorphic Encryption is that you can create your report without taking the documents out of the safe.

Additive Property of Homomorphic Encryption:
A Homomorphic encryption is additive, if $Ek(PT1 \oplus PT2) = Ek(PT1) \oplus Ek(PT2)$

Multiplicative Property of Homomorphic Encryption:
Homomorphic encryption is multiplicative, if $Ek(PT1 \otimes PT2) = Ek(PT1) \otimes Ek(PT2)$

Algorithm	Additive	Multiplicative	Applications
RSA	No.	Yes	To secure Internet Banking and credit card transactions
Paillier	Yes	No	E-voting system
ElGamal	No.	Yes	In Hybrid Systems



Complexity of Multiplication Algorithms

Grade School Method	Karatsuba Method	Fast Fourier Transform
$O(n^2)$	$O(n^{\log_2 3}) = O(n^{1.58})$	$\Theta(n \log(n) \log(\log(n)))$

RSA Algorithm

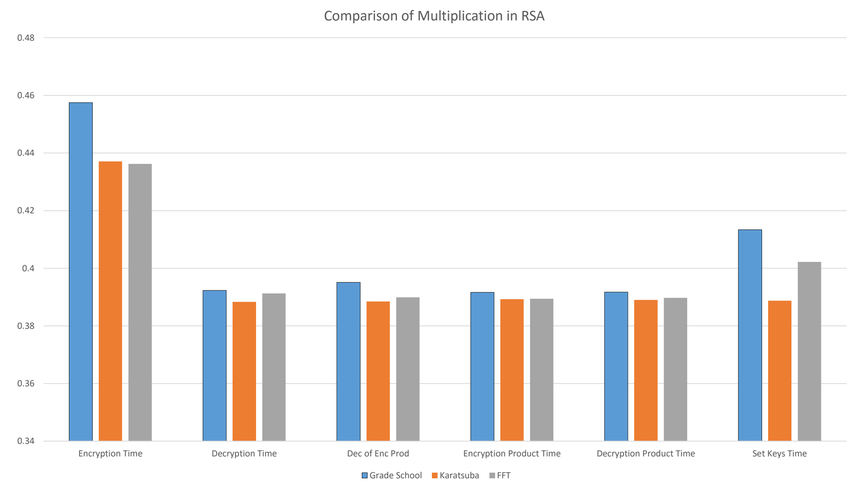
- Selecting two large primes at random: p, q
- Computing their system modulus n=p.q
- Note ø(n)=(p-1)(q-1)
- Selecting at random the encryption key e where 1<e<ø(n), gcd(e,ø(n))=1
- Solve following equation to find decryption key d
- e.d=1 mod ø(n) and 0≤d≤n
- Publish their public encryption key: pu={e,n}
- Keep secret private decryption key: pr={d,n}
- To encrypt a message M the sender:
  obtains public key of recipient pu={e,n}
  computes: $C = m^e \bmod n$, where $0 \le m < n$
- To decrypt the ciphertext C the owner:
  uses their private key pr={d,n}
  computes: $M = c^d \bmod n$

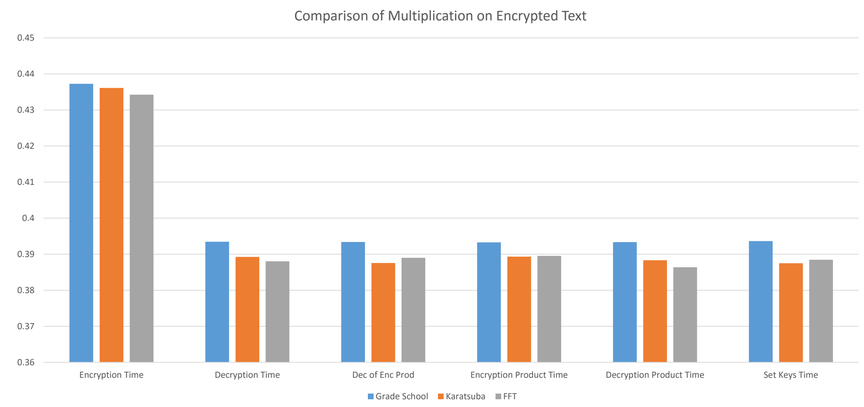
Implementation

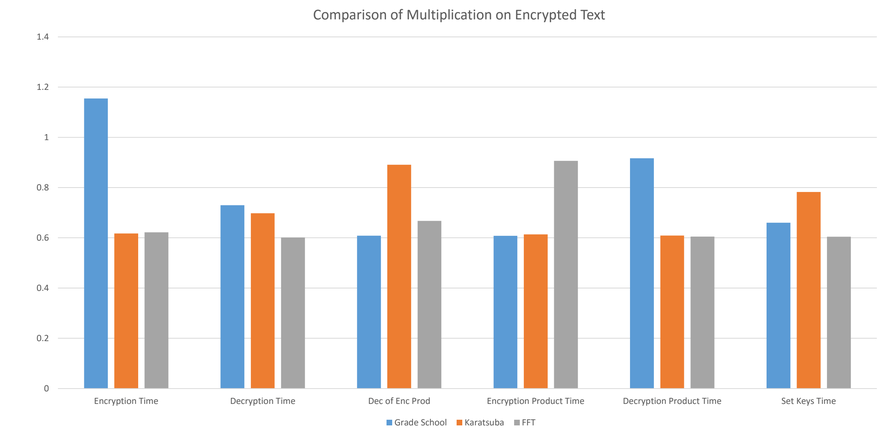
The implementation of algorithms is performed in two steps
1. Multiplication algorithms in RSA algorithm were replaced by Karatsuba and FFT methods one after another.
2. Multiplication operations were performed on encrypted numbers by Karatsuba and FFT Methods.
3. Each of the Multiplication algorithm and RSA algorithm was is implemented as SPADE agent.

In both of the above cases algorithms were tested on Laptop with 8 GB RAM, i5 Processor, Ubuntu 19 OS and Virtual Machine with Ubuntu Mint, 1.5 GB RAM and i5 processor.

Results







Conclusions

- Homomorphic Encryption enables computation on untrusted resource. The Computation time over cipher text can be reduced by using Karatsuba and FFT methods.
- Need to test the computation time with respect to homomorphic properties of Elgamal, Paillier and ECC Systems.
- Multi agent system can be used for load balancing

References

1. William Stallings,Cryptography and Network Security: Principles and Practice, Pearson, 7th Edition

Acknowledgements

I thank Scipy Organizers for awarding scholarship to attend Scipy 2019. I also thank KLS Gogte Institute of Technology, Belagavi for providing me an opportunity to represent the institution and present my work.