A Python based Approach to Enhance Security using Secret Sharing Scheme

Siyaram Gupta
Department of Information Technology
Uttarakhand Technical University,
Dehradun, Uttarakhand, India

Madhu Sharma
Department of Computer Science Engineering
DIT University,
Dehradun, Uttarakhand, India

ABSTRACT
Privacy of the information is one of the major issues in digital world. However it is responsibility of the owner of the information to secure it from disclosure and unauthorized access. The main motivation behind this protocol is to ensures individual privacy along with integrity of the information. To achieve such security requirements a sharing scheme formally known as secret sharing scheme is proposed. According to this well-known (k, n) threshold scheme at-least k shareholders must gather to regenerate the secret [2]. The Lagrange interpolation polynomial used to construct the shares for participants [3] in this proposed concept. Secret sharing is a method for distributing a piece of secrets amongst a group of individuals, each of which is allotted some share related to the secret. The secret can only be reconstructed according to its threshold value. Python is used to implement the concept of secret sharing scheme because of its ease of use and huge collection of functions, libraries and modules. It is easy to generate and use polynomials using python.

Keywords
Cryptography, Secret sharing, Polynomials, Python.

1. INTRODUCTION
Security and protection of information is a major issue in almost every field using insecure channels for transmission of data among different parties. Such concerns led us to use cryptographic techniques to maintain the confidentiality of information, it also assure that information should not be tempered in any way during its transmission. In cryptography, secret sharing is a scheme through which a secret or information is distributed among a number of parties. The major issue is that how to distribute or share secrets in such a way that only authorized person can access. Shamir's scheme divides the information into n pieces so that each participant gets their share. Each participant also has a unique value for their identification as valid participant. The secret is revealed if and only if k participants must gather k-1 participants do not allow recovering the secret. In general, in this scheme there are n players and one dealer, the dealer is responsible for distribution of shares but only when some predefined conditions are fulfilled. The dealer use such scheme by distributing the share in such a way that the particular group of 't' or more members which has the share can together recover the secret but no group which has less than 't' members can able to access the secret. The scheme is named as Shamir's secret sharing scheme or a (t, n) - threshold scheme.

Processing and handling information by computers and sharing them over high-speed network infrastructure have become a common practice since wide deployment of low cost computing and networking hardware. Currently, student/teacher text files and records are stored on disks of college/university database systems for fast and reliable storage and retrieval. Protection of the integrity and confidentiality of department information is an issue in the management of student/teacher personal & professional records. Confidentiality states that unauthorized parties should not be granted to access those records during transmission. Integrity, on the other hand, implies that information such as results should not be modified in any way during transmission.

These schemes use polynomial interpolation for shares. There are k points in the 2-D plane (X_i, Y_i), …… (X_k, Y_k) with unique X_i's values, there is one polynomial h(x) of degree (k-1). Modular arithmetic is more suitable than real arithmetic for such purpose. Pick an integer prime number P in this way that is bigger than the secret S and the number of members' n. The (k, n) threshold scheme has some properties such as the size of each share does not exceed the secret size. Share S_i can be added or deleted dynamically once k is kept fixed. Change in S_i pieces does not affect the original secret anymore. These properties of the scheme enhance security since exposed pieces by security holes can't be accumulated until the values of all are from same edition of the polynomial. We can get a hierarchical scheme as S is determined by number of shares depends on importance. This is a hierarchical scheme and each share has its own important and values.

2. RELATED WORK
To keep the secret confidential and provide privacy of information, Shamir (1979) [2] and Blakley [4] first developed the concept of secret sharing scheme. There are lots of applications and areas that implement the concept of secret sharing. The scheme is highly ideal for encryption keys, missile codes, bank account, hiding medical images and patient's records etc.

However, as these schemes are discussed, there are several drawbacks such as limited number of secrets can be shared during sharing process , dishonest dealer , malicious participants may distribute fake shares to other shareholders. Such issues are further fix and improved by many researchers. Multi-secret sharing (MSS) [11-12] schemes have been proposed to overcome the drawback that shares only single secret at a time. C.C. Yang, T.-Y. Chang, M.S. Hwang [6] proposed a new MSS, which is based on the two-variable one-way function [5] in 2004. To overcome the problem of dishonest dealers and malicious participant scholars proposed

the verifiable secret sharing (VSS) schemes. AVSS scheme allows participants to verify the validity of shares of the other participants and her/him. The first realization of VSS was written by Chor et al. [7] in 1985.

Thereafter, Harn [8] presented the verifiable multi-secret sharing (VMSS) scheme in 1995. Some researchers improve this scheme but the cost of computing in it is still high. The YCH scheme is a relatively efficient multi-secret scheme at the present time. But the issue is that the scheme doesn't have the property of verification.

A new literature [9] has a property of verification based on YCH scheme; the scheme is unpractical because of its big-ticket system. To overcome the drawback of the scheme ,later a new practical verifiable multi-secret sharing scheme proposed by Jianjie Zhao ,Jianzhong Zhang ,Rong Zhao [3] which is based on discrete logarithm [10] and intractability.

3. OVERVIEW OF SHAMIR'S SHARING SCHEME AND PYTHON

3.1 Overview of Shamir's sharing scheme

Shamir's secret sharing scheme divides the information into n pieces so that each participant gets their share. Each participant also has a unique value for their identification as valid participant. The secret is revealed if and only if k participants from pool of n must gather, k-1 participants do not allow to recovering the secret. The scheme proposed by Shamir is based on Lagrange interpolation polynomial; polynomial concept is used to construct shares for participants.

The scheme divides the secret S into n shares denoted by (S_1, S_2, ... S_n). A (k-1) degree polynomial with a large prime number P is used to compute shares as in (1).

$$P(x) = (S+a_1 x+a_2 x^2+\cdots+a_{k-1}x^{k-1}) \bmod M \qquad (1)$$

(a_1, a_2, ⋯, a_{k-1}) are the coefficient of the polynomial selected randomly from within range (0, M]. The computed shares are as in (2).

$$Y_1 = (1, P(1)), Y_2 = (2, P(2)), Y_3 = (3, P(3)). \ldots\ldots\ldots\ldots\ldots\ldots\ldots Y_n = (n, P(n)). \qquad (2)$$

The shares are pair of two integers. If k of pairs gathers, then only participants are able to reconstruct the secret using Lagrange's interpolation technique. The constant term S of the polynomial is the secret. This scheme is suitable for secret sharing.

3.2 Overview of python

Python language is used for the implementation of sharing scheme. Python is an interpreter, object-oriented, high level scripting and programming language. Guido van Rossum was the first who introduce the python language in 1991**.** Python is very easy to learn and use and it's a open source language, it helps a lot at the time of implementation. A generation of polynomials is an easy task in python language. Python is also useful for us to generate random numbers. We are using GMPY for multi-precision in this scheme. It provides a great platform for its users; the manuals and documents provided by python help the beginner. It has almost all libraries and functions a user needs to implement his work. Numpy, gmpy, flint etc are the most common functions and libraries used during implementation. It has effective approach to object-oriented programming and high level, efficient data structure. Features like elegant syntax and dynamic typing make it one of the powerful language and because of its interpreted nature, python is used as scripting language; it is also useful in development of applications for most of the platforms. The data types and functions implemented in C or C++ (Cython) Java in the form of Jython are easily used in python. Python extends there features.

Following are the main features and functions that are used during implementation:-

Random Function: Used to generate random numbers such functions are "random.randrange ()" and "random.getrandbits ()".

Example: a =random.getrandbits (64)

(Generate a random number of 64 bits.)

GMPY for multi-precision: A C-coded Python extension module that supports multiple-precision arithmetic. Gmpy support for the MPFR and MPC libraries. The gmpy mpz type supports arbitrary precision integers. Gmpy provides a rational type call mpq. Gmpy uses mpf or mpfr type to support multiple-precision reals.

"gmpy.mpz ()" is for multi-precision integer value.

Example: *import gmpy*

y=gmpy.mpz (64)

(A 64 bit gmpy integer variable)

Numpy Library: Python provide numpy library that is useful for numeric calculations. NumPy (Numeric Python) package provides basic routines for Array creation routines, Linear Algebra, Polynomials, Random numbers and much more.

Example:	*import numpy as npp*
Array Creation	*a = npp.array ([1, 4, 5, 8], float)*
Polynomials	*npp.poly ([-1, 1, 1, 10])*
Random Numbers	*npp.random.rand (5)*

There are lots of applications of python such as scripting language for web applications, Libraries like Matplotlib, Gmpy, SciPy and Numpy are the libraries that allows python to be used effectively and easily in complex and scientific computing. Natural language processing tasks and artificial intelligence works are also done by using python. Applications such as medical image security, EPR hiding [1] implements this scheme to keep their records secret.

4. PROPOSED SCHEME

Shamir's sharing scheme demonstrates how a secret can be shared so that the secret is recovered only when sufficient number of shares is available. This scheme gives basic logic of a secret being shared among five people. Out of these five people, if at least three people give their shares the secret can be regenerated.

4.1 Secret sharing using python

Thus in the Shamir's (n, k) we take (5, 3)**:**

The proposed scheme uses quadratic polynomial since it has three coefficients. The three coefficients are the secret to be shared. The five shares are generated by evaluating the coefficients (Y_s) at randomly chosen X_s. Thus (X[i], Y[i]) make up the i^{th} share. Only when at least three such shares are available the original polynomial can be regenerated.

All calculations are done modular taking the mod as My_mod = gmpy.mpz (2^{64} - 59), which is the largest 64-bit prime number. The mod has to be a prime as otherwise the modular arithmetic will become very tedious and impractical. It has two functions – Quad_Poly_Eval and Quad_Poly_Regen for evaluating and regenerating a quadratic polynomial. Org_Poly is the original polynomial and Rec_Poly is the regenerated polynomial. The shares are randomly generated.

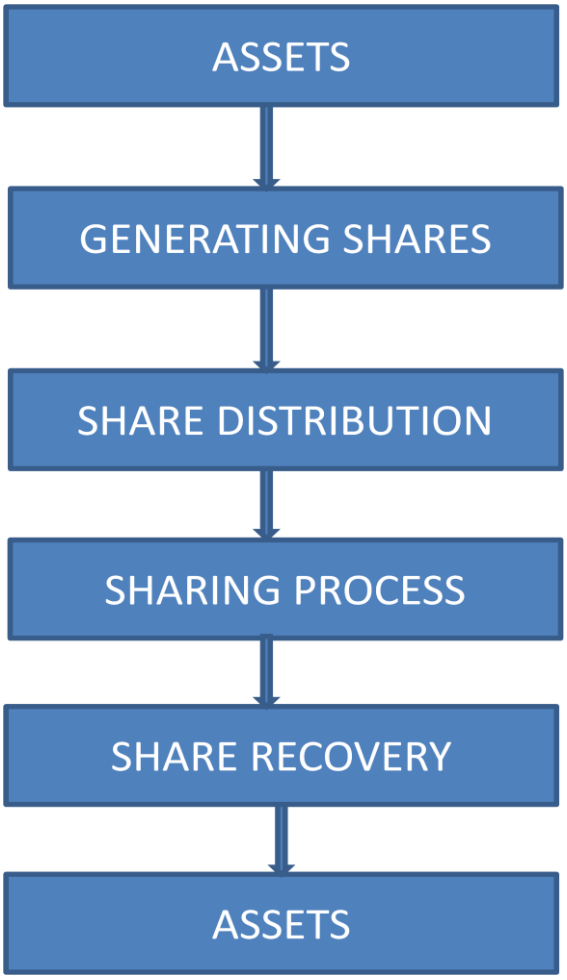

Fig 1: Secret Sharing Process

4.2 Implementation of proposed scheme

The Scheme has two main phases:

4.2.1 Share Construction Phase:

My_mod = gmpy.mpz (2^{64} - 59)
(Largest 64-bit prime number.)

Quad_Poly_Eval (coeffs, x, mod)
(This evaluate quadratic modular polynomial defined by coeffs at x, mod is the modulus.)

Org_Poly = [gmpy.mpz (random.randrange (0, My_mod))]
(Org_Poly is the evaluated or original quadratic polynomial.)

X_s = [gmpy.mpz (random.randrange(0, My_mod))
(Randomly chosen value.)

Y_s = [Quad_Poly_Eval (Org_Poly, X_s [0], My_mod)
(Generate share for each participants.)

4.2.2 Share Recovery Phase:

Quad_Poly_Regen (Xs, Y_s, mod)
(Finds modular quadratic polynomial passing through points given by X_s & Y_s).

Rec_Poly = Quad_Poly_Regen ([X_s [0], X_s [1], X_s [2]], [Y_s [0], Y_s [1], Y_s [2]], My_mod)
(Rec_Poly is the recovered quadratic polynomial.)

Org_Poly is the original polynomial and Rec_Poly is the regenerated polynomial calculated using Quad_Poly_Regen function.

$ ipython

Python 2.7.5 (default, Jan 10 2014, 09:40:52)

In [1]: execfile ("mypolym.py")

In [2]: Org_Poly

Out [2]:
[(8963919782422927 38), (3706908100683070 7926L), (9726351682425429984)]

In [3]: Rec_Poly

Out [3]:
[(8963919782422927 38), (3706908100683070 7926L), (9726351682425429984)]

Out [2] and Out [3] are the outputs of Org_Poly and Rec_Poly respectively. Both outputs show same results, so by using the concept of polynomial and sharing scheme it is easy to evaluate and regenerate shares.

If a part, say X_s [1] and Y_s [1] is repeated, an error occurred and the polynomial is not regenerated:

In [4]: poly1 = Quad_Poly_Regen ([X_s [0], X_s [1],

X_s [1]], [Y_s [0], Y_s [1], Y_s [1]], My_mod)

ZeroDivisionError Traceback (most recent call last)

<ipython-input-5-dbafdc471c87> in <module> ()

----> 1 poly1 = Quad_Poly_Regen ([X_s [0], X_s [1], X_s [1]], [Y_s [0], Y_s [1], Y_s [1]], My_mod)

/home/ash/myresearch/Cryptography/mypolym.py in Quad_Poly_Regen (X_s, Y_s, mod)

28 for j in range (0, 3):

29 if (j != i):

*---> 30 coeffspart [2] *= gmpy.divm (1, X_s[i] – Xs[j], mod)*

31 coeffspart [1] -= X_s[j]

*32 coeffspart [0] *= - X_s[j]*

ZeroDivisionError: not invertible

Also, if a part, say Y_s [3] in place of Y_s [2], is repeated, an error occurred and the polynomial is not regenerated:

In [5]: poly2 = Quad_Poly_Regen ([X_s [0], X_s [1], X_s [2]], [Y_s [0], Y_s [1], Y_s [3]], My_mod)

In [6]: poly2

Out [6]: [(1314589107488295383 8L),

(543580718118861428), (3960423652794329163)]

In [7]:

5. CONCLUSION AND FUTURE WORK

The proposed paper adopts the concept of secret sharing first introduced by Adi Shamir. The concept has the property of sharing and securing the intellectual work of the people. The main aim of such a concept is to ensure and satisfy the security of information and to fix privacy issues. The scheme prevents unintentional disclosure of information to those who are not allowed to access it. Secret sharing scheme is a very powerful concept of sharing. The scheme assures privacy, integrity and authentication of the information. Polynomial coefficients used in this secret sharing scheme provides secrecy of the important information. Implementation of this scheme using python makes it more secure and powerful. Further the proposed scheme will be enhanced by using some cryptographic techniques. By using encryption technique the shares should be more secure and the scheme will be more powerful.

6. REFERENCES

[1] Medical image security and EPR hiding using Shamir's secret sharing scheme, Mustafa Ulutas, Güzin Ulutas∗, Vasif V. Dept. Of Computer Engineering, Karadeniz Technical University, 61080 Trabzon, Turkey.

[2] A. Shamir, How to share a secret, Communications of the ACM 22 (11) (1979) 612–613.

[3] A practical verifiable multi-secret sharing scheme. Jianjie Zhao, Jianzhong Zhang, Rong Zhao. College of Mathematics and Information Science, Shaanxi Normal University, Xi'an 710062, People's Republic of Chin b Natural Science Institute, Xi'an University of Technology, Xi'an 710048, People's Republic of China.

[4] G. Blakley, Safeguarding cryptographic keys, Proc AFIPS 1979 National Computer Conference, AFIPS Press, New York, 1979, pp. 313–317.

[5] J. He, E. Dawson, Multisecret-sharing scheme based on one-way function, Electronics Letters 31 (2) (1995) 93–95.

[6] C.-C. Yang, T.-Y. Chang, M.-S. Hwang, A (t, n) multi-secret sharing scheme, Applied Mathematics and Computation 151 (2004) 483–490.

[7] B. Chor, S. Goldwasser, S. Micali, B. Awerbuch, Verifiable secret sharing and achieving simultaneity in the presence of faults, Proc. 26th IEEE Symp. FOCS, 1985, pp. 251–260.

[8] L. Harn, Efficient sharing (broadcasting) of multiple secret, Computers and Digital Techniques 142 (3) (1995) 237–240.

[9] J. Shao, Z.-F. Cao, A new efficient (t, n) verifiable multi-secret sharing (VMSS) based on YCH scheme, Applied Mathematics and Computation 168 (2005) 135–140.

[10] R.-J. Hwang, C.-C. Chang, An on-line secret sharing scheme for multisecrets, Computer Communications 21 (13) (1998) 1170–1176.

[11] H.-Y. Chien, J.-K. Tseng, A practical (t,n) multi-secret sharing scheme, IEICE Transactions on Fundamentals of Electronics, Communications and Computer 83-A (12) (2000) 2762–2765.

[12] J. He, E. Dawson, Multistage secret sharing based on one-way function, Electronics Letters 30 (19) (1994) 1591–1592.