A Versatile and Ubiquitous Secret Sharing

A cloud data repository secure access

[1]Muhammad Adeka, [1]Simon Shepherd, [2]Nuredin A. S Ahmed, [1]Raed Abd-Alhameed
[1]School of Electrical Engineering and Computer Science, Faculty of Engineering and Informatics
University of Bradford,
Bradford, West Yorkshire, BD7 1DP, United Kingdom
{M.I.Adeka@student., S.J.Shepherd@, R.A.A.Abd@}Bradford.ac.uk
[2]Faculty of Engineering, Azzaytuna University, Tarhuna, LIBYA

***Abstract* - The Versatile and Ubiquitous Secret Sharing System, a cloud data repository secure access and a web based authentication scheme. It is designed to implement the sharing, distribution and reconstruction of sensitive secret data that could compromise the functioning of an organisation, if leaked to unauthorised persons. This is carried out in a secure web environment, globally. It is a threshold secret sharing scheme, designed to extend the human trust security perimeter. The system could be adapted to serve as a cloud data repository and secure data communication scheme. A secret sharing scheme is a method by which a dealer distributes shares of a secret data to trustees, such that only authorised subsets of the trustees can reconstruct the secret. This paper gives a brief summary of the layout and functions of a 15-page secure server-based website prototype; the main focus of a PhD research effort titled 'Cryptography and Computer Communications Security: *Extending the Human Security Perimeter through a Web of Trust*'. The prototype, which has been successfully tested, has globalised the distribution and reconstruction processes.**

Keywords – authentication; secret sharing; cryptography; key management; interpolation; authorised user; human security perimeter; (k, n)-threshold; participants (trustees); dealer or distributor; combiner; cloud data repository

I INTRODUCTION

Adeka's Versatile and Ubiquitous Secret Sharing System (*AdeVersUS*[3]), a cloud data repository secure access, is a web based authentication scheme. It is designed to implement the sharing, distribution and reconstruction of a sensitive secret data; e.g., a combination key for firing a nuclear missile, the access code for a flow station, the Coca-Cola formula or any highly sensitive data that could compromise the functioning of an organisation, if leaked to unauthorised persons. This is carried out in a secure web environment, globally. It is a threshold secret sharing scheme, designed to extend the human trust security perimeter. Though primarily designed as a secret-sharing system, it could be adapted to serve as a cloud data repository and secure data communication scheme. A secret sharing scheme is a method by which a dealer (appointed by an organisation) distributes pieces (shares) of a secret data to a group of people (trustees, participants or players), such that only authorised subsets of the trustees can reconstruct the secret. Secret sharing schemes are important tools in cryptography which are used as a building block in many secure protocols; e.g., general protocol for multiparty computation, Byzantine agreement, threshold cryptography, access control and attribute-based encryption. This paper gives a brief summary of the layout and functions of a 15-page secure server website prototype; the main focus of a PhD research effort titled 'Cryptography and Computer Communications Security: *Extending the Human Security Perimeter through a Web of Trust*' [1].

The paper will briefly highlight the theoretical basis for secret sharing, a summary of design and implementation of *AdeVersUS*[3] prototype, the practical results, discussions and conclusions with recommendations for future work.

II THEORETICAL BASIS FOR SECRET SHARING

In cryptography, *secret sharing* is a method by which a given *secret* is distributed among a *set* of *participants* (*trustees*), each of whom is given only a *share* of the secret. Reconstruction of the secret would only be possible when all the participants or a stringently defined minimum *subset* of participants (*access structure* or *authorised set*) pool their shares together. In other words, both individual shares and any number of shares less than the authorised set are of no use on their own. Thus, generally, the access structure of a (k, n)-threshold scheme normally partitions the set of all subsets of participants into *authorised sets* who can recover the secret and *unauthorised sets* who cannot; some schemes feature an intermediate third class of subsets, who are neither authorised nor unauthorized [2]. The aim of the scheme is to provide tight control over the sensitive data and remove single-point vulnerability.

The objective of a secret sharing scheme is to make a given secret *D* (say some data, e.g., key combination) inaccessible to unauthorized persons, while making it accessible to authorized persons when the need arises. It is assumed that non-mechanical solutions which could manipulate this data in the process are allowed [3]. The goal is to divide *D* into *n* pieces $D_i, \ldots, D_n$ such that:

- Knowledge of any *k* or more D_i pieces makes *D* easily reconstruct-able.
- Knowledge of any *k-1* or fewer D_i pieces leaves *D* completely indeterminable (in the sense that all its possible values are equally likely; assumed absolute randomness).

A Key Recovery via Threshold Schemes

Threshold schemes are often referred to as secret sharing or secret splitting. A secret (such as a session key or private/public key pair) is split into several shares, a subset of which must be combined to recover the secret. For instance, a secret might be split into 6 (*n*) shares and any 3 (*k*) might be needed to recover it. The value *k* in the above scheme is called the threshold number, while *n* is the share size. Any reasonable share size and recovery threshold is possible (provided the threshold number is less than or equal to the share count). In order that the secret data or key be recoverable, it must be an RSA private key **[4]**. The RSA cryptographic algorithm consists of three main steps; namely, key generation, encryption and decryption. While the public key can be known to everyone and is used for encrypting messages, all messages encrypted with the public key can only be decrypted with the use of the private key, which is kept secret and known only to its owner, the originator. RSA derives its security from the difficulty of factoring large prime numbers. Its public and private keys are functions of a pair of large prime numbers (100 to 200 digits or larger). A decryption of the ciphertext using the private key is perceived as being equivalent to factoring the product of the two large prime numbers [5].

B ***Mathematical Definition***

The main idea behind Shamir's Secret Sharing Scheme (SSSS) is based on polynomial interpolation. [3] The polynomials could be replaced by any other functions which are easy to evaluate and to interpolate. This idea is rooted in the notion that two points are enough to define a line, three points are required to define a quadratic expression, four points are required to define a cubic function and so on. In other words, it requires 'k' points to define a polynomial of order 'k-1' [6].

Given *k* points in the Cartesian plane $(x_1, y_1),\ldots,(x_k, y_k)$ with distinct x_i's, there is one and only one polynomial g(x) of order *k-1*such that $g(x_i) = y_i$ for all *i*. Without losing generality, it can be assumed that the data D is a number or it could be made a number. In order to divide it into pieces D_i, pick a random *k*-1degree polynomial

$$g(x) = a_0 + a_1x + \ldots a_{k-1}x^{k-1} \quad . \quad . \quad . \quad . \quad . \quad . \quad (2)$$

where $a_0 = D$. Then evaluate:

$D1 = g(1), \ldots, D_i = g(i), \ldots, Dn = g(n).$

Using any subset of k of the D_i values, the coefficients of g(x) can be found by interpolation, and then evaluate $D = g(0)$. Knowledge of just k-1 of these values, on the other hand, is not enough to calculate *D*. In order to make this claim more precise, [3] uses modular arithmetic {i.e., finite field arithmetic or arithmetic in the Galois Field [*GF(p)*]} instead of real arithmetic; the set of integers modulo a prime number p forms a field within which interpolation can be carried out.

- *Example*

In a threshold scheme, a secret data (a prime number) is shared among 5 trustees. The key can be recovered by using any 2 shares. Show that the secret can be recovered from any 2 of the following shares and hence determine the secret data. (Consider the use of graphical observation/interpolation; hence, start by plotting the points to solve the problem).

$S_1 = (-3, 13)$, $S_2 = (-1, 9)$, $S_3 = (2, 3)$,
$S_4 = (4, -1)$, $S_5 = (6, -5)$.

Plotting the points (S_i) as given will produce the graph in Figure 1. It is clear that any two points are sufficient to reproduce the line since all of the points above lie on it. Thus, the secret data, represented by the point at which the graph intersects the *y*-axis, is clearly 7.

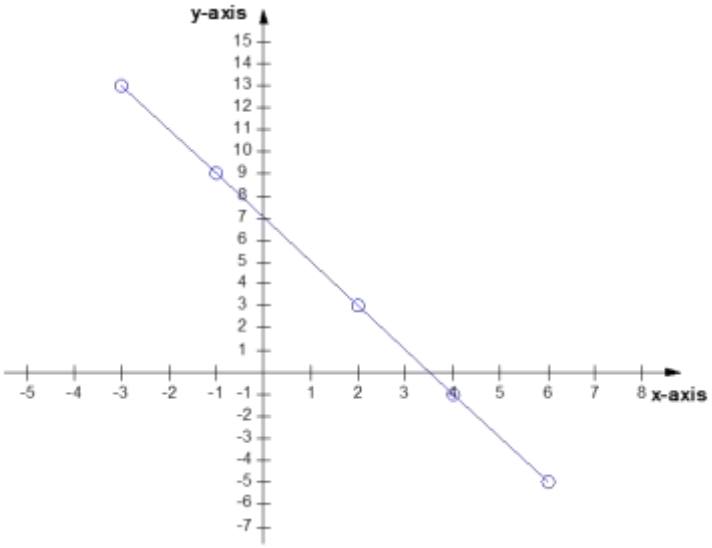

Fig. 1. Determining the Secret Data for a Straight Line Graph.

Mathematically, the above result can be verified as follows. Using the general formula for a line:

$y = mx + c$, where m is the gradient = (change in y)/(change in x) between two points

Take two points on the line e.g. (4, -1) and (6, -5)

The gradient m is $(-5 - (-1)) / (6 - 4) = - 4/2 = -2$.

Using the point (4, -1) and substituting $m = -2$ into the above equation gives:

$-1 = -2(4) + c$

Therefore,

$c = 7$ (same as determined graphically above).

III PROTOTYPE DESIGN AND IMPLEMENTATION

A Web Design Requirements

The design envisaged the engineering and development of a website whose web pages would display a real-time digital clock (showing the GMT or UTC timing), and the site would require at least more than one form of authentication for access, for security reasons. It would also display a timer (down counter).

Supposing that a secret item (e.g., a password) is shared among five participants who are scattered around the world; say one each in Russia, China, Nigeria, USA, UK and Australia, and each of them has only a fraction or share of the Secret, which, on its own, is grossly insufficient to determine the secret. It is required that at least three or more fractions of the password domiciled in any three or more of the above identified countries around the world must be assembled in order to reconstruct the whole password. Inherent in these requirements are the authentication needs, each, to determine the identity of each participant, his/her geographical location and the authenticity of his/her share or fraction of the Secret to be reconstructed.

In the event that the password in use by the rightful owner or authority is missing or gets destroyed, some of the participants with fractions of it would be required to recombine or reconstruct it from their different locations around the world without coming together geographically. Each of them would be required to enter his/her own

fraction of the password on the website within a specified short period of time (say 5 minutes; e.g., between 1420 and 1425 hours GMT of a given date) so that the entire password would be reconstructed at a designated GPS location, as may be directed or determined by the Dealer (coordinator or administrator). After reconstruction, only the designated/authorised participant (s) should be able to access the data; this accessibility should be possible once only.

B Web Design Implementation

The *AdeVersUS*[3] prototype utilises Shamir's Secret Sharing Scheme (SSSS), MD5 encryption and user registration methods to securely create and share data through a web browser in a Wi-Fi configuration; or using local host in a wired LAN. The system is built up using HTML5, PHP, Java, Servlets, JSP, MySQL, JQuery, and CSS. These are running on Tomcat and Apache data bases using XAMPP Server. There are various libraries used such as the Database Connector, Joda Time, Google's JSON parser and Shamir's Algorithm. The source code is object oriented and adheres to software engineering tools and principles. In order to run the server via the local host: start up the Tomcat server (Java) in Eclipse, Apache (PHP) and MySQL (database). Point the browser to http://localhost:8080/secretshare/ (NB: *secretshare* is the project file name which could be changed any time). Each user (participant) must be registered to access the system. When the user registers, the password and GPS coordinates could be encrypted using MD5. The user must also give his GSM contact information at the time of registration

The prototype website contains about 15 web pages. These include the Admin, Login, Secure Share, and Secret Data web pages. Apart from issues relating to connectivity and database, some of the prototype functions are also represented in the Use Case Diagram, Sequence Diagram and State Diagram. The basic flow of the prototype functional states is shown in the state diagram in Figure 2. The web development followed the design steps for the Waterfall model.

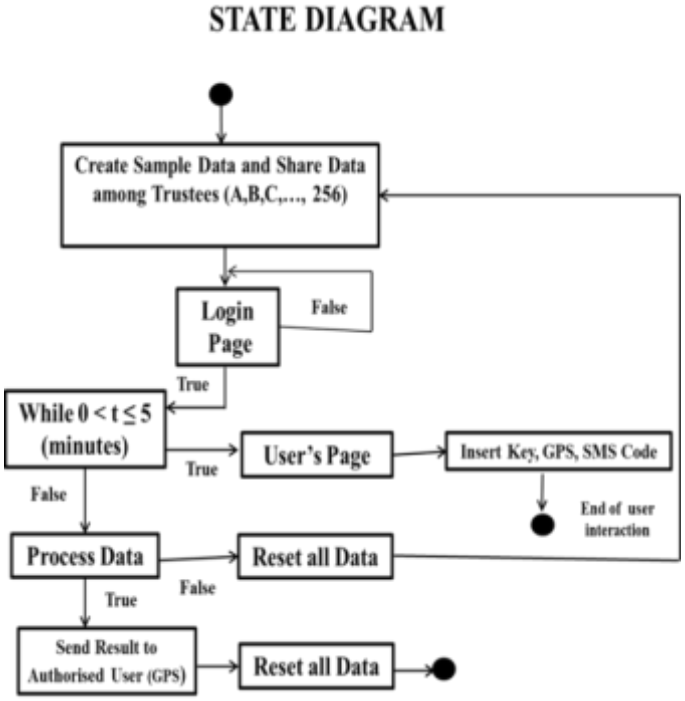

Fig. 2. AdeVersUS[3] Prototype State Diagram

IV PRACTICAL RESULTS AND DISCUSSIONS

This segment highlights how the AdeVersUS[3] prototype actually functions and the novelties or areas of new knowledge that it has been able to accomplish in the science of secret sharing till date. The discussion ends with a basic comparison of the Prototype system and the SSSS, using *security* and *time* as metric parameters. The net diagram of the real AdeVersUS[3] prototype is in Figure 3.

A Admin Page

The Admin web page can only be viewed by the user on the server side within the local host; any external connections are denied. This page must be fully configured before a share distribution session can commence. For instance, if the start time on the Admin page has not been set, then the user cannot login. When the Admin sets the start time and the time duration is specified, the end time which is automatically calculated from these is also displayed. The default format for HTML5 date-picker is used for the date and user display time. All the values can be set as in Figure 4; this illustrates a threshold quorum of three out of five shares. It is also on this page that the sharing process can be restarted to begin another session, after a successful recovery of a secret data in the previous session, or to repeat the previous session in the event of a failure. Point 3 allows the user who is authorised to view the data to be selected. Point 8 allows the keys to be assigned to specific users; in a given situation where users are not serially assigned according to the database serialisation.

Fig. 3. Adeka's Global Secret Sharing Scheme Employing a Secure Server at the School of Electrical Engineering and Computer Science, Faculty of Engineering and Informatics, University of Bradford, United Kingdom, with a Client each at Los Angeles, USA; Rio de Janeiro, Brazil; Abuja, Nigeria; Canberra, Australia; Beijing, China; and Moscow, Russia.

Fig. 4. Part of Page 1 of Admin Page.

Page 2 of the Admin web page (Master Share Details) displays part of the results of the settings configured in Figure 4; this is to assist while testing the prototype. Once the form in Figure 4 is submitted, the system sends the information to Shamir's Secret Sharing Algorithm. This encrypts the data and returns a number of keys based on the setting above. The system uses HTTP GET request to use SSSS API. The system uses the GSON library to parse the data into a DTOSecret object. A sample JSON is:

{"timeStart":"1368221578","secretKeys":["0101596f75"],"numberOfShares":"1","quorum":"1"}.

As part of the user data displayed on Page 2 of the Admin Page as determined by the setting in Figure 4, this is a sample response from the SSSS algorithm, where it has split the secret data into a single key; as set in the Admin. After this stage, all the data needed for a session is set and the system is ready to handle users and share secret data.

B Login Page

The opening web page (home page) on the client side allows the user to log in and register; he registers only for the first time, which could be edited afterwards, if he wishes. Here the user input is entered through the hashed keypad. Once the login has been submitted, using a POST request through HTTP, it gets passed through the MD5 function and compared with the MD5 value stored in the database. In order to prevent an SQL Injection Attack, the system uses prepared statements [7]. In logging in, for example, security verification checks are carried out to prevent security breaches: e.g., does the entered name exist and does the md5 password match? A 'Yes' leads to the opening of the next web page; the Secure Share page. The Login page uses a hashed map key pad as shown in Figure 5, where JQuery is used to get the key pad values and input them into the text box when the text field is clicked.

C Secure Share Pages

The Secure Share Page is where the users log into to add their keys, GPS data and mobile authentication code. Similar verification checks are again carried out, using MD5 hashes and other data in the database, as done in the login page, to ensure that the data added meet the criteria. That is, does the GPS match; has the correct user entered the correct key assigned to him; and does the verification code sent to the user match? The page also makes provision for editing of personal data, if the user wishes to do so. Part of the Secure Share web page is shown in Figure 6. The result of a successfully authenticated entry by a user into a share/reconstruction session for all users, except the authorised user, is shown in Figure 7; that of the authorised user is as in Figure 8. The prototype system generates a Universally Unique Identifier (UUID) and sends it to the mobile number registered by the user for additional authentication purposes. The SMS message contains the pass code for the data share. This is implemented using a built-in java UUID generator. The Secure Share page has a countdown clock, which expires based on the share duration set in the Admin page. Once all the shares (or the required threshold) have been added, the data can be unlocked. The authorised user gets to see a page with a countdown and a button to reveal the secret data. Once the authorised user clicks the View Data Button (which appears only on his own screen), various checks are carried out; to ensure that each user has added the key and code assigned to him correctly, his GPS data is correct, the quorum (threshold) has been met and if within the time limit; i.e., the countdown clock is still counting as at the time the last user successfully enters his data. Finally, the keys are sent back into the SSSS algorithm for final check.

D Secret Data Viewing Page

If all went well without any error, a successful sharing process would display the Secret Data page when the authorised user clicks the View Data Button; otherwise, an appropriate error message is displayed on the page.

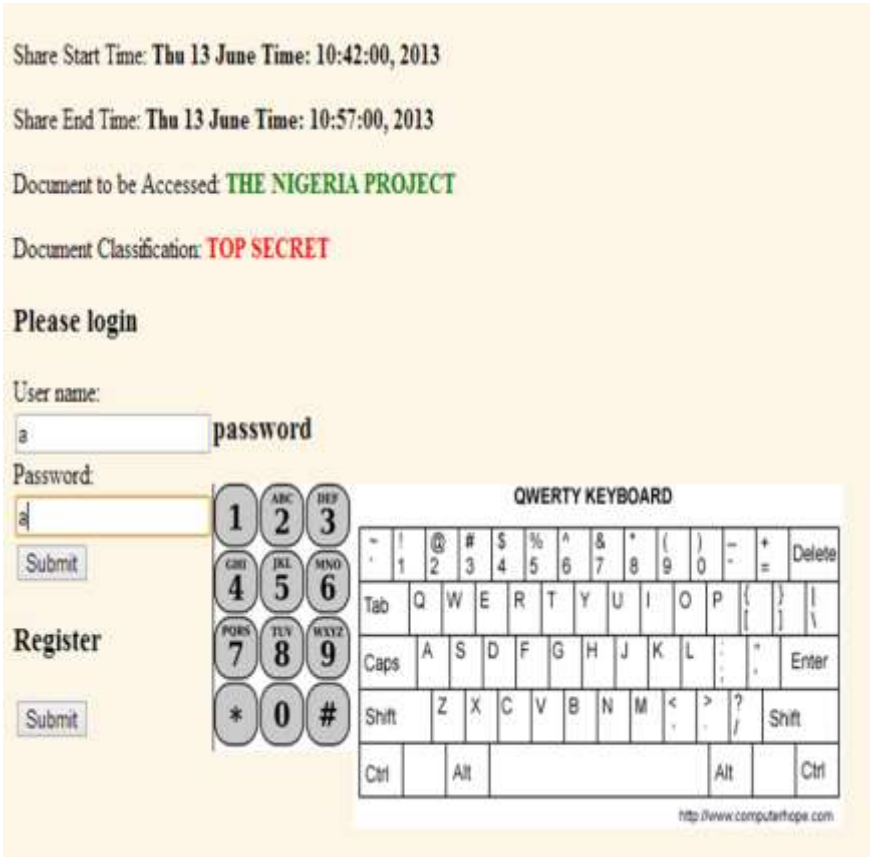

Fig. 5. Part of Page 1 of Login Page: Login Form.

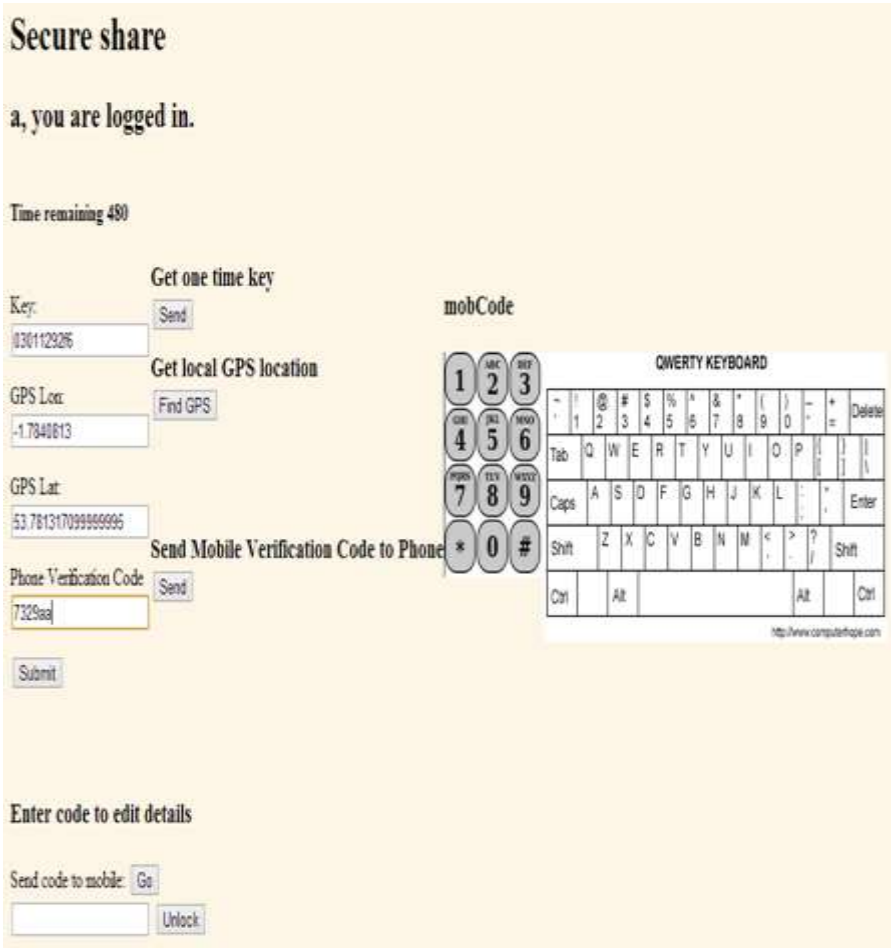

Fig. 6. Secure Share Page

If the display shows success, the system is restarted, ready for the next share session; if it shows a message that indicates failure, the system is restarted, ready to repeat the failed session. The authorised user can view the reconstructed secret data once only. The system gets wiped out as soon

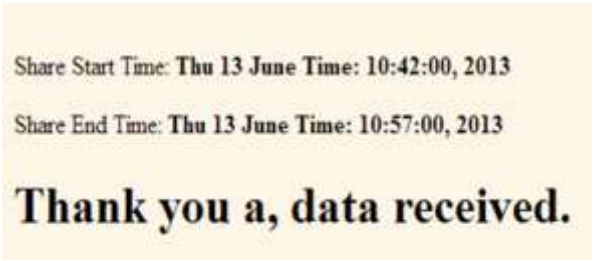

Fig. 7. Response for a Successful Entry by a User (Participant).

Fig. 8. Response for a Successful Entry by the Authorised User.

as the secret data has been unlocked and displayed. Any page refreshment or re-attempt will result in an error message. A sample result of this page is shown in Figure 9.

Fig. 9. Secret Data Page Displaying Success

E Connectivity

The system currently runs on a laptop via local host within a typical home router, using Wi-Fi. Remote access to this system can be achieved by setting up port forwarding on the router. This is required because the IP address in the

the server will have an internal IP address: 192.168.x.x (which can be viewed in the command prompt by typing *ipconfig*). This IP needs to be registered in the router settings for port forwarding. Once this is set, the router will forward any requests on port 8080 (port could be different) to the local host on 192.168.x.x:8080/secretshare/ - which is the login page. The next stage is to find the external IP address of the local router, which can be obtained via a simple Google search. Then any remote users can point their browsers to <router-ip-address>:8080/secretshare/ and start using the sharing system. The server can also be remotely accessed through a wired LAN with an appropriate switch. The prototype has been successfully demonstrated using both techniques. Using port forwarding as described above, the server can be accessed from anywhere in the world.

F Areas of New Knowledge

The main novelties that have been accomplished so far relate to modifications, in form of additions, in the SSSS algorithm, in an effort to resolve some of the identified weaknesses in the (k, n)- threshold schemes. These include: The geographical spread of trustees which is now global in nature as against a one-location based recombination process; Due to its globalisation, location-based user authentication techniques are introduced – e.g., employment of the GPS coordinates and SMS text mobile authentication codes; the inclusion of a dynamic time window, digital clock and timer; other long-standing unresolved issues in relation to secret sharing, which have now been resolved in the AdeVersUS[3] include answers to the questions - Who is the *Combiner*, Where should the recombination take place and Who is entitled to have access to the reconstructed secret? It is also noteworthy that the practical implementation of a web based authenticated secret sharing scheme, with all the complements of the AdeVersUS[3], has no precedence.

G Comparing AdeVersUS[3] with the SSSS

It would not be fair to make any reasonable claims of its performance characteristics in comparison with the SSSS algorithm, as some *analo-synthesists* (analysts[+] and synthesists[-]) might be erroneously tempted to do. This is

because, while the SSSS is an algorithm for breaking up a secret data into pieces and reconstructing same, AdeVersUS[3] prototype is a pragmatic service, which practically implements the secret sharing and reconstruction scheme globally; using the Shamir's algorithm as a cryptographic primitive, along with various other building blocks and incorporated modern authentication techniques. Comparing the SSSS algorithm with AdeVersUS[3] prototype secure server is akin to a comparison between the vacuum tube diode/semi-conductor transistor and a radio transceiver system.

V CONCLUSION

AdeVersUS[3] employs various building blocks, including the SSSS algorithm, MD5 and various libraries. Its programing elements include HTML5, PHP, Java, Servlets, JSP, MySQL, JQuery, and CSS. These are running on Tomcat and Apache data bases using XAMPP Server within the Eclipse IDE. The prototype system does work and the data can be shared across the globe via a secure server based website. It has passed JUnit tests, but has not been tested under heavy traffic. The phone number is registered during the registration process but the user is allowed to edit it along with other existing user details; these are potential areas of security concern subject to modifications. Hacking into the system to illegally access the data would be a very tough task. There are many checks to ensure that every condition has been met before the data is unlocked. SMS-text-verification-based authentication gives an added level of security and a modern touch. The same is true of GPS coordinates. The prototype achieved some novelties as enumerated in Section IV (F).

Further work would focus on the following: having a managed area to set up future shares at a given period; having more than one different share sessions happening simultaneously; capturing all the user interactions in a database; other incorporations aimed at making the system more robust and versatile; and subjecting the system to various performance tests. In addition, instead of splitting the secret data itself as keys to be shared among participants, it might be better to share randomly generated keys which are then used to protect the secret data. This would be a more secure approach in the event that the on-going efforts at homomorphic encryption become a reality.

ACKNOWLEDGMENT

The roles of the Petroleum Technology Development Fund (PTDF, Nigeria) for sponsoring the PhD Programme that produced this work, and the Nigerian Army for granting approval for the research, are hereby acknowledged. I am greatly indebted to my supervisors and other staffs of the School of Electrical Engineering and Computer Science, as well as fellow research colleagues, especially Mr Kelvin Anoh, and the Library, ICT and other facility services at the University of Bradford, UK. Thanks to my wife, Saudat, and all my children/wards. Glorified is Allah, the Lord of His Creation.

REFERENCES

[1] M. Adeka, S. Shepherd, and R. Abd-Alhameed, "Cryptography and Computer Communications Security: Extending the human security perimeter through a web of trust," *Unpublished PhD Research Thesis Draft, School of Electrical Engineering and Computer Science, Faculty of Engineering and Informatics, University of Bradford*.

[2] K. M. Martin, "Challenging the adversary model in secret sharing schemes," *Coding and Cryptography II, Proceedings of the Royal Flemish Academy of Belgium for Science and the Arts*, pp. 45-63, 2008.

[3] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, pp. 612-613, 1979.

[4] F. M. Reza, *An introduction to information theory*: Courier Corporation, 1994.

[5] T. Rid, "Cyber war will not take place," *Journal of Strategic Studies*, vol. 35, pp. 5-32, 2012.

[6] C. L. Liu, *Introduction to combinatorial mathematics* vol. 181: McGraw-Hill New York, 1968.

[7] W. G. Halfond and A. Orso, "AMNESIA: analysis and monitoring for NEutralizing SQL-injection attacks," in *Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering*, 2005, pp. 174-183.