Enhancing Totp Protocol By Embedding Current Gps Location

U. A. Abdurrahman, M. Kaiiali and J. Muhammad

***Abstract*—Static password authentication is no longer considered secure in the internet and banking services. This is because, easy-to-guess passwords such as first-six-digit numbers, names and date of birth are easily uncovered by automated password stealing devices or programs. Multi-factor authentication has been proposed to meet the demand of organizations by providing stronger authentication options to their users. Time-based One-Time Password (TOTP) protocol is one of the widely accepted mechanisms that incorporate multi-factor authentication. TOTP is adopted in Google Authenticator and Windows Authenticator software. However, this protocol has certain limitations; it uses a static shared key and stores it in a clear form into SQLite database (DB) on the mobile device; the timestamp used as a seed to generate the OTP is easy-to-guess; and its mobile app is not PIN protected. Due to these limitations, there is a need to provide a countermeasure to these issues. This paper proposes an enhancement to TOTP by introducing GTOTP. GTOTP generates the OTP based on pre-shared key, timestamp and user's current GPS location. The use of GPS location increases the strength of the generated token by embedding dynamic information (GPS) into the seed used to generate the OTP. The paper discusses different attack scenario to prove that the proposed GTOTP is secure. GTOTP has been implemented and tested. Difficulties of incorporating GPS values into the seed have been addressed.**

***Index Terms*—Multi-Factor Authentication, TOTP, Web Security, Hashing, GPS Applications**

I. Introduction

Today, computing becomes universal, people now are relying on computers to do their business. The internet has become a medium for uncountable e-services on which people perform their business transactions. However, there is a need to provide an utmost protection to the system, the information being sent and the network used to process the information.

Information security refers to the provisions and policies adopted to prevent and monitor unauthorized access, misuse, modification or denial of a computer network and network-accessible resources [1, 2]. It also involves the authorization of access to data in a network. Users are assigned an ID associated with a chosen password or other authenticating information that allows them to access information and programs within their authority [1].

Traditionally, users tend to use static passwords in securing their systems. However, these passwords suffer potential attacked risks: passwords can be guessed, forgotten, stolen, eavesdropped or deliberately being told to other people. The solution to these problems is by the use of a more secure way of authentication called multi-factor authentication [3]. A multi-factor authentication technology is used to provide higher security guarantee than static passwords.

A multi-factor authentication is an approach for authentication, which requires the use of two or more of the universally recognized authentication factors [4] (as illustrated in Fig. 1):

1) A *knowledge* factor: something you know (e.g. password, PIN)
2) A *possession* factor: something you have (e.g. smart cards, tokens)
3) An *inherence* factor: what you are (e.g. biometrics)
4) An optional fourth factor *relatives*: who you know (e.g. brother, wife).

Fig. 1. Multi-factor authentication [5].

So many methods of multi-factor authentication have been presented in the literature in which so many researchers made use of a combination of two or more of the above mentioned factors. One of the multi-factor authentication approaches that is widely in use is SMS-based authentication, where a randomly generated One Time Password (OTP) is sent by

Manuscript received July 30, 2014.

Abdurrahman U. A. is an Assistant Lecturer at the Department of Computer Science, Northwest University, Kano, Nigeria (e-mail: usmanalhaji79@gmail.com).

Mustafa Kaiiali is an Assistant Professor at the Department of Computer Engineering, Mevlana University, Konya, Turkey (e-mail: mkaiiali@mevlana.edu.tr).

Muhammad J. is a Research Assistant at the Department of Computer Engineering, Mevlana University, Konya, Turkey (e-mail: mohdjawadi@yahoo.com).

service providers to the user's mobile phone through an SMS message [6]. This requires additional cost for sending SMSs to all customers who are willing to login. Unlike SMS-based multi-factor authentication, the proposed GTOTP authentication does not require any additional cost.

Nystrom proposed a method of multi-factor authentication using a SecurID which is based on a security device [7]. In this method, each device has a unique seed used to generate a pseudo-random number (OTP) to authenticate the user along with his other credentials. The device is timely synchronized and its seed is stored in the server in order to validate the sent OTP. RSA SecurID [8] is an example of this method (Fig. 2). This approach has an advantage over the SMS-based one as it does not require additional cost for sending SMSs and it is accessible worldwide, unlike SMS-based approach which is limited within the geographic boundaries of a particular country unless the international roaming service is activated. However, for large companies specifically those who offer free services such as Yahoo and Google, it is costly and infeasible to deliver the security device to each customer. Moreover, the device itself may got stolen.

Compared to SecurID, the proposed GTOTP is considered to be more secure as SecurID is based on a fixed seed while the proposed GTOTP will be based on a modifiable pre-shared seed and dynamic GPS location. Moreover, GTOTP is cheaper and easier to deploy, as it does not require a specific security device like SecurID. Nowadays, due to the revolution in smartphone technology, our phones have sufficient applications, gadgets and sensors that can be utilized for secure authentications rather than using specific devices (i. e. SecurID). This reduces the cost and increases the usability.

Fig. 2. RSA SecureID Authentication [9].

Liou, Jing, and Sujith proposed a method similar to SecurID called SofToken that generates a pseudo-random number as an OTP based on a seed shared between the authentication server and a software application installed on the user's computer/mobile [10]. The seed is renewed for every successful login. This approach eliminates the cost of the hardware token; however, it increases the security risk as the new shared seed is sent over non-secure channel so it can be intercepted by an intruder.

Another software based OTP generator has been proposed in [11, 12]. It uses IMEI and IMSI numbers as the seed to generate the OTP. However, IMEI number is not very secure as it is known to the mobile service provider; also IMSI is a cellular network identifier which makes the method not valid worldwide (in case if the user travels worldwide and changes his SIM card accordingly).

Sabzevar, Alireza and Angelos stated a method of using a graphical image to represent a password; the password is entered by pointing on appropriate points of the image, which the user has obtained through his mobile device from the service providers [13]. It has the same drawback of the previous approach as it also depends on particular service provider.

In another paper by Soleymani and Maheswaran who-you-know-based authentication factor was used as the second factor of authentication [14]. This factor is used as a voucher, where a user asks a friend to vouch for him. The user contacts his friend asking for a vouching code which he then uses to log-in to the web page. They make use of user's cellphones to automate this process. Each time a user calls a friend, a token will be generated (vouching) for this contact. A sufficient number of these tokens can be used to authenticate a user. However, this method complicates the authentication process.

Another popular method of authentication currently in use is HOTP/TOTP-based authentication [15]. In this method, an internet user can install an Authenticator App on his mobile phone. This app computes the verification code using a HMAC-based OTP (HOTP) or Time-based OTP (TOTP) as shown in Fig. 3.

Fig. 3. HOTP/TOTP-Based Authentication by Google [16].

With HOTP, the server and client share a secret value and a counter, which are used to compute an OTP independently on both sides. Whenever a password is generated and used, the counter is incremented on both sides, allowing the server and client to remain in sync. This algorithm works based on an increasing counter value (C) and a fixed symmetric key (K) shared between the token generator and the validation system as shown in (1). The Authenticator App uses HMAC-SHA-1 algorithm for the generation of HOTP value [17].

$$\text{HOTP}(K, C) = \text{Truncate}(\text{HMAC-SHA-1}(K, C)) \qquad (1)$$

Where the Truncate function extracts 4-byte dynamic binary code out of a 160-bit HMAC-SHA-1 result and then reduces it to modulo 10^{digit}.

On the other hand, TOTP uses the same algorithm as HOTP with one major difference. The counter used in TOTP is replaced by the current time as shown in (2). The client and server remain in sync as long as the system times remain the same. This is done by use of Network Time Protocol. The shared secret key is generated by the server and delivered to the client using a form of QR Code [18].

$$\text{TOTP} = \text{HOTP}(K, T) \tag{2}$$

K is the shared key, T is an integer representing the number of time-steps between the initial counter time T_0 and the Current UNIX Time. T is computed as shown in (3) where X is the time-step in seconds (30 sec by default).

$$T = (\text{Current UNIX Time} - T_0) / X \tag{3}$$

The advantage of this algorithm is that user does not have to be connected with any kind of service provider (Internet, GSM, etc.) to get the verification codes. However, despite being one of the most popular and widely accepted authentication mechanisms, TOTP implementations, such as Google Authenticator and Windows Authenticator, have the following weaknesses:

1) It uses a static pre-shared key and stores it in a clear form into an SQLite DB in the mobile device.
2) The dynamic part of the seed used to generate the OTP is only the timestamp, which is easy-to-guess.
3) The mobile app used to generate the OTP is not PIN protected. So it can be broken if somebody stoles the mobile device.

This paper proposes an enhancement to the TOTP protocol (named as GTOTP) by embedding current GPS location in the seed used to generate the OTP. Embedding GPS location in the seed makes it more dynamic and so the OTP will be more secure and harder-to-guess. Table I compares the proposed GTOTP with the previous discussed multi-factor authentication mechanisms.

TABLE I
COMPARING THE DISCUSSED MULTI-FACTOR AUTHENTICATION METHODS

	SMS-Based	SecurID	TOTP	GTOTP
Cost	Not Free	Not Free	Free	Free
Service Access	Restricted	Worldwide	Worldwide	Worldwide
Secure Seed	Fixed	Fixed	Dynamic / Easy to Guess	Dynamic / Hard to Guess
Service Provider	Cellular Network	Not Required	Not Required	Internet
Stolen Device	Not Protected	Not Protected	Not Protected	Protected

The remaining sections of this paper are organized as follows: Section II presents the proposed GTOTP authentication mechanism. Section III discusses the GTOTP implementation issues. Section IV provides a security analysis of the proposed protocol. Section V concludes the paper.

II. GTOTP Authentication Mechanism

GTOTP has five main components: user's initial registration component, a client-side application that runs on smartphones, a database for storing user's standard credentials, a server-side authentication service and a GPS server to track user's GPS location. All of these components are synchronized using NTP Protocol. The GTOTP scheme adopts a user's GPS location in addition to pre-shared key and timestamp to strengthen the security of the generated token. Unlike the static pre-shared key and the easy-to-guess timestamp, GPS location is dynamic, hard-to-guess information that will increase the security of the generated hash.

A. Initial Registration (To be done once)

One of the critical parts that need to be addressed properly in the aforementioned protocol is how to negotiate the shared keys between the user's phone (client) and the server. GTOTP adopts the TOTP way of sharing the key. However, GTOTP requires two shared keys, one to be used as a seed for OTP generation (as in TOTP) and the other to be used for establishing secure channel with the GPS server discussed later. To securely negotiate a shared key, here are the steps:

1) The user should enable the multi-factor authentication service from his/her account settings page.
2) The server generates the shared key, stores it associated with current user id and presents it on the user's screen in barcode form (Fig. 4 shows the barcode that corresponds to the key: **MA4Q-EUH5-BA7U-XYZC**).
3) The user clicks on "scan barcode" button in his/her client app. The phone's default barcode scanner will be launched. The user positions the phone's camera at the barcode appears on his/her computer screen in order to scan it. By doing so, the GTOTP client app learns and stores the shared key.

Fig. 4. Secret Shared Key embedded in a bar code.

B. GTOTP Authentication

After negotiating the secure channel parameters between the authentication server and the user's mobile device. The user can go ahead to access his account any time he/she wants according to the following three steps illustrated in Fig. 5.

1) Initial Authentication:

In this step, the user requests his/her personal web page URL through any internet-enabled device D_1 as shown in Fig. 5. The Web Portal Server S_1 has to respond back to D_1 with the authentication page asking the user to provide his/her traditional credentials (username/password). The user provides his/her credentials to D_1, which sends them to S_1 for verification. If the user is verified, the system proceeds to the next step.

2) GTOTP Security Token Generation

The GTOTP client app, installed on the user's mobile device D_2, is used to generate the GTOTP security token based on the following parameters:

- The Pre-Shared Key: It is a secret key which has previously been shared between the client and the server during the initial registration process. This parameter is inherited from TOTP
- Timestamp: This is the current time in D_2. It helps to randomize the seed used to generate the GTOTP One Time Password. It also reduces the risks of replay attacks. This parameter is also inherited from TOTP.
- GPS Location: User's current GPS location is another important factor used to strengthen the security token. This factor is unique to the user and very dynamic. While TOTP seed is based on static key and easy to guess timestamp, GPS location increases the seed dynamism and makes it harder to guess. Current GPS location is synchronized with the GPS server through a pre-established secure channel (whose key was negotiated during the initial registration). Therefore, the GPS server keeps track of user's location every time the security token is generated.

All the above-mentioned factors are concatenated to form GTOTP seed. GTOTP keeps using HMAC-SHA-1 algorithm to generate its OTP. HMAC-SHA-1 acts upon the seed and produces a 160-bit message digest. This message digest is further truncated to 6-digits for easy entering by the user. This process results in a security token T_1 (4) that is unique and last for 30 seconds period.

$$T_1 = \text{Truncate (Hash (Pre-Shared Key} \parallel \text{GPS} \parallel \text{TS))} \qquad (4)$$

At the same time D_2 updates the GPS server (S_2) with the user's GPS location and timestamp (TS) through a secured channel.

3) Token Verification

S_1 receives the security token T_1 and gets the GPS updates of the corresponding user from S_2. Following that, it generates a security token T_2. Then it compares T_2 with the one it has received from the user (T_1) as shown in Fig. 6, if they match, the user is authenticated and his/her personal web page is sent back to D_1.

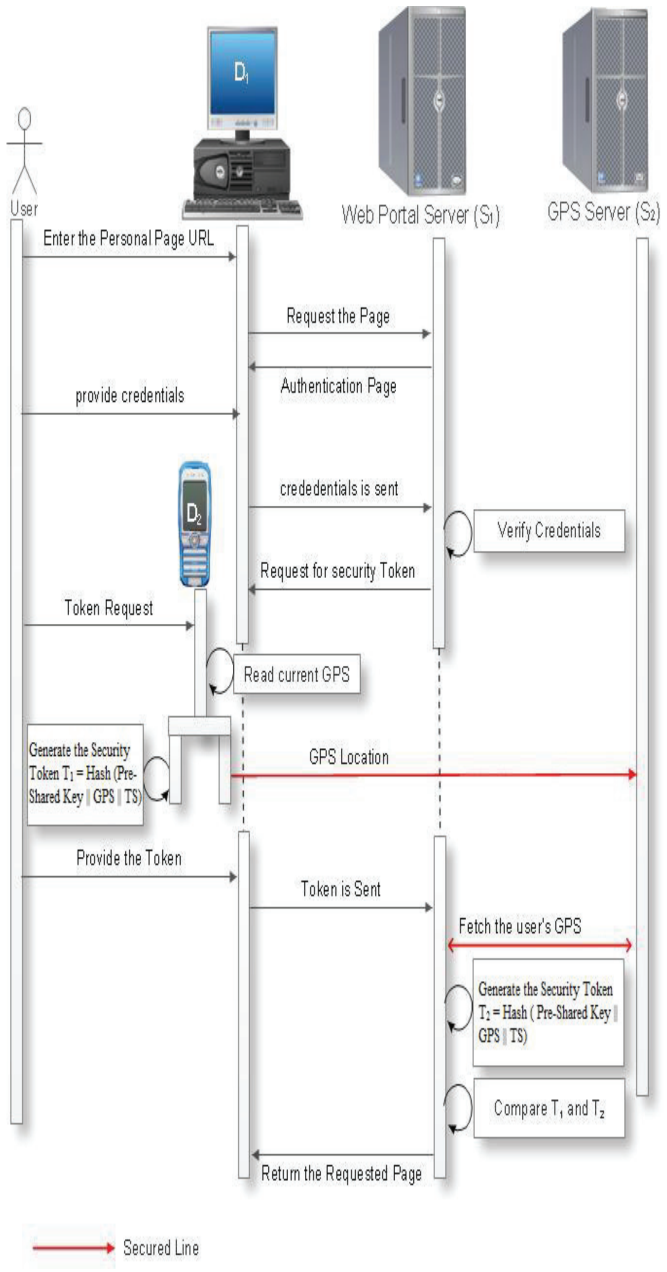

Fig. 5. GTOTP Authentication [3].

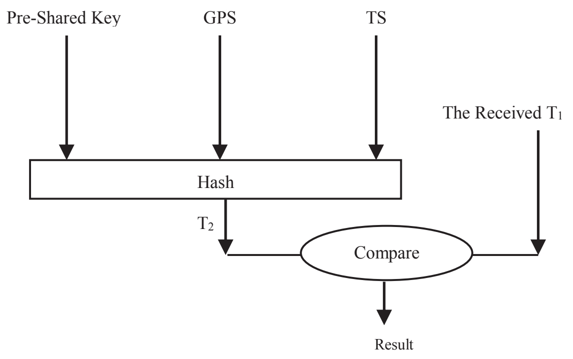

Fig. 6. Token Verification [3].

III. GTOTP Implementation Issues

A. *Device Theft Countermeasure:*

Many TOTP implementations such as: Google Authenticator are not protected against device theft. For that, we have developed the GTOTP client app to be PIN protected in order to mitigate the threat of shared key disclosure by mobile device theft.

B. *Shared Key Protection:*

TOTP implementations such as: Google Authenticator stores the Pre-Shared Key in a clear form in an SQLite database. This increases the chance of key disclosure using malicious software accessing the database. For that, GTOTP client app stores the shared key encrypted using the user's app PIN as well. Once the app is launched, the user is prompted to provide the PIN, and then the app fetches the encrypted key from the SQLite database and decrypts it using the provided PIN.

C. *DYNAMISM:*

A shared key = ZO5UJAY5RMH2E72U, a timestamp = 12569537309, and 10 different GPS locations (of approximately 1 km difference each) have been considered and the security token has been computed and depicted in Table II. The experiment shows that for a very small standard deviation (STD) of the GPS values, we get a large STD for the generated security tokens. This shows that for a little change in user's GPS location, there will be a drastic change in the security token. This proves the dynamic effect of the GPS coordinates on the generated token.

TABLE II
Standard Deviation of 10 GPS Locations and Their Correspondent Security Tokens For a Fixed Timestamp and Shared Key

	Latitude	Longitude	Security Token
Location 1	23.001	32.01	527273
Location 2	23.002	32.02	238761
Location 3	23.003	32.03	579027
Location 4	23.004	32.04	246264
Location 5	23.005	32.05	577068
Location 6	23.006	32.06	624019
Location 7	23.007	32.07	361999
Location 8	23.008	32.08	299430
Location 9	23.009	32.09	639977
Location 10	23.01	32.1	014026
Standard Deviation	0.002872	0.028723	199092.9

D. *ACCURACY:*

The user's GPS coordinates changes significantly only when he/she changes his/her position to an approximately 10 meters [21]. The dynamism of GPS appears significantly if the user changes his location for more than 10 meters.

E. *LATENCY:*

The higher the number of satellites a GPS receiver can view the better the accuracy [19]. However, due to so many factors, like atmospheric effect, multipath effect, etc. a GPS signal reception can be delayed or blocked. Moreover, GPS units typically do not function underwater, indoors, or underground [20]. These factors certainly affect the accuracy of the GPS signals and consequently increase the GPS latency.

For these reasons, our GTOTP implementation is designed to use the previous GPS location of the user to generate the security token in case it finds it difficult to receive a new GPS signals within 30 seconds.

F. *PRIVACY:*

Using GPS location as a parameter in generating the security token raises the issue of revealing user's location without his/her consent. To overcome this issue, our GTOTP implementation is designed in such a way that, the use of GPS parameter in generating the security token is made optional. GTOTP app gives the user the option to exclude GPS location from the OTP seed; in this case, the app works exactly as the earlier discussed TOTP code generator. The only difference here is that our app is PIN protected and the shared key is stored encrypted.

G. *Lost Mobile Device:*

If the user has lost his/her mobile device, he/she can no longer access his/her personal web page. To encounter this issue, GTOTP provides an option to pre-register an alternative mobile device (let's say user's wife mobile device) that can be used alternatively in this case. Probably the server has to ask the user extra security questions before it allows him/her to use the pre-registered alternative mobile device.

IV. GTOTP Security Analysis

This section shows how the proposed scheme can resist various threats:

A. *Resistance against D1 device hack:*

If an intruder gains access to the user system (D_1) and obtain information about his standard credentials (username and password). The intruder still won't be able to log in to the server as he does not have the information required to generate the GTOTP token.

B. *Resistance against replay attack:*

An intruder obtained information about user's credentials (username and password) and he was also able to intercept the communication between the mobile device and the authentication server to capture the sent token. With the token being made to get expired by time (i.e. 30 seconds) and by first use (valid to be used for one time only), the intruder won't be able to use the captured token even within its lifetime as it has already been used by the user.

C. *Using one-way hash function:*

Assuming the intruder has the user's fixed credentials (username and password) and he/she was also able to know the user's current GPS location (being his/her colleague). Still he/she cannot generate a valid token as he does not have the

Pre-Shared Key. The Pre-Shared Key has never been transmitted over any channel during authentication. Since the used hash function is a one-way and pre-image resistance function, there is no way for the intruder to get the Pre-Shared Key out of a sniffed token.

D. *Using PIN protected app:*

If an intruder has stolen the user's mobile phone and has already obtained the user's fixed credentials (username and password), for this scenario, we made the GTOTP client app a PIN protected app, unlike Google Authenticator. Therefore, the intruder would not be able to use the app as far as he does not know its PIN.

V. Conclusion And Future Work

This paper highlights the TOTP limitations and addresses them by introducing an enhanced protocol called GTOTP. It incorporates the GPS parameter in the seed, which makes it more dynamic and increases the strength of the generated token as well. So many multi-factor authentication approaches have been discussed in the literature. However, each of them has certain security issues in one way or another: either it is easy to compromise; or difficult to implement; or requires an additional cost of obtaining a special device. Our GTOTP makes use of mobile phone (that is common to all users) to generate security tokens (OTPs). This method has been effectively implemented on Android, tested and proved to be robust and secure. An overview of the various parts of the system has been provided in the paper. Additionally, a security assessment was also carried out.

As for further work, the GTOTP algorithm can be extended to include user's biometric parameter to increase the security of the system. Also GTOTP has to be implemented on Blackberry, Apple and Windows mobile phones to provide a wider range of support.

References

[1] "Network Security", http://en.wikipedia.org/wiki/Network_security

[2] User, Assurance, and Audit Planning. "An Introduction to Computer Security: The NIST Handbook." (1995).

[3] Abdurrahman, U. A., Kaiiali, M., & Muhammad, J. (2013, November). A new mobile-based multi-factor authentication scheme using pre-shared number, GPS location and time stamp. In *Electronics, Computer and Computation (ICECCO), 2013 International Conference on* (pp. 293-296). IEEE.

[4] Bauckman, Dena Terry, Nigel Paul Johnson, and David Joseph Robertson. "Multi-Factor Authentication." U.S. Patent No. 20,130,055,368. 28 Feb. 2013.

[5] "Secure Your Future Now", http://info.softexinc.com/bid/54466/Data-Breach-should-you-be-concerned

[6] Alzomai, Mohammed, Bander AlFayyadh, and A. Josang. "Display security for online transactions: SMS-based authentication scheme." *Internet Technology and Secured Transactions (ICITST), 2010 International Conference for*. IEEE, 2010.

[7] Nystrom, M. "The SecurID (r) SASL Mechanism." (2000).

[8] "SecurID", http://en.wikipedia.org/wiki/SecurID

[9] "RSA's Recent Compromise", http://www.thewestbrooks.com/bruce/techblog/2011/03/21/409

[10] Liou, Jing-Chiou, and Sujith Bhashyam. "A feasible and cost effective two-factor authentication for online transactions." *Software Engineering and Data Mining (SEDM), 2010 2nd International Conference on*. IEEE, 2010.

[11] Aloul, Fadi, Syed Zahidi, and Wassim El-Hajj. "Two factor authentication using mobile phones." Computer Systems and Applications, 2009. AICCSA 2009. IEEE/ACS International Conference on. IEEE, 2009.

[12] Eldefrawy, Mohamed Hamdy, Khaled Alghathbar, and Muhammad Khurram Khan. "OTP-Based Two-Factor Authentication Using Mobile Phones."*Information Technology: New Generations (ITNG), 2011 Eighth International Conference on*. IEEE, 2011.

[13] Sabzevar, Alireza Pirayesh, and Angelos Stavrou. "Universal multi-factor authentication using graphical passwords." *Signal Image Technology and Internet Based Systems, 2008. SITIS'08. IEEE International Conference on*. IEEE, 2008.

[14] Soleymani, Bijan, and Muthucumaru Maheswaran. "Social authentication protocol for mobile phones." *Computational Science and Engineering, 2009. CSE'09. International Conference on*. Vol. 4. IEEE, 2009.

[15] "TOTP", https://tools.ietf.org/html/rfc6238.

[16] Borchert, Bernd, and Max Günther. "Indirect NFC-Login."

[17] "HOTP", https://tools.ietf.org/html/rfc4226

[18] "Google Authenticator", http://code.google.com/p/google-authenticator/wiki/KeyUriFormat

[19] Zhang, Pei, et al. "Hardware design experiences in ZebraNet." *Proceedings of the 2nd international conference on Embedded networked sensor systems*. ACM, 2004.

[20] Tan, Hwee-Pink, et al. "A survey of techniques and challenges in underwater localization." *Ocean Engineering* 38.14 (2011): 1663-1676.

[21] LaMarca, Anthony, et al. "Place lab: Device positioning using radio beacons in the wild." *Pervasive computing*. Springer Berlin Heidelberg, 2005. 116-133.