



An HOTP Based Algorithm to Enhance Wi-Fi Security

Rohit Tolani , Mrs. Anjali Yeole, Mr. Sachin Gavhane

Department of Computer Engineering, V.E.S.I.T.,
Mumbai, India

Abstract- This paper aims to enhance the security of Wi-Fi, one of the most prevalent media for internet access. With a view of providing city-wide wireless connectivity, a new algorithm based on the HMAC based One-time password (HOTP) algorithm is proposed. In places of public access, using your static passwords is compromising with your own security. One-Time Passwords provide random passwords that are harmless even if captured by sniffers. The new algorithm accommodates these requirements, and more, to secure user credentials in public networks. The algorithm has been evaluated on parameters such as image size, password size and period of repetition based on security vs. overhead of generation.

Keywords- HMAC based OTP, HOTP, One time password, Public Wi-Fi access, Steganography, Wi-Fi security

I. Introduction

With the dawn of the smart-world where almost everyone must have access to a smart phone or a feature rich gadget, Wi-Fi has seen a boon as a means to stay connected over the internet. In the last decade, the rapid rise of network threats has exposed the inadequacies of static passwords as the primary mean of authentication. To improve user security in cyberspace, multilevel authentications combined with biometrics have been devised. Newer concepts like city-wide Wi-Fi access plan to provide Wi-Fi to everyone on the go. Since wireless networks are easily accessible than wired networks and there are a plethora of ways to hack and crack Wi-Fi passwords, it is necessary to have more added security to protect Wi-Fi access. As a step further to the standard “static” password authentication for Wi-Fi networks, an algorithm based on HOTP, which generates dynamic passwords, is being proposed. This algorithm uses Lamport’s Mechanism to generate new OTPs from the old OTP. An added module of random noise image generation uses steganography to secure the masterkey of the user, enabling him to expose only his OTP which is not vulnerable to replay attacks. Modern cryptography does not have an absolute algorithm. Thus, any algorithm is a trade-off between strength, speed and other features such as scalability and flexibility. Hence, an analysis of the various parameters has been presented.

II. Inception

Pablo Vidales et al [1] in providing a solution for Metropolitan Public Wi-Fi Access Based on Broadband Sharing, proposed the idea of Extended HotSpots. This was based on the IEEE 802.11 technology. They have provided a business perspective on the discussed models and they also reported the results of the field trial in Berlin. Although Extended HotSpots is an efficient solution, the ISP still needs to verify users and validate their passwords. A more efficient solution would be to generate dynamic passwords for public users who access the private networks occasionally. The users can save their static passwords at their home networks and use dynamic passwords in public networks. Gamal Hussein [2] in his analysis of Randomized Cryptosystems Attacks and Defenses showed that if the OTP is exchanged securely with the user, dynamic modelling of a two stage random number generator (TSRG) achieves provable security based on insoluble problem with respect to the attacker. A detailed study of the three one-time password generation techniques facilitated the selection of Lamport’s mechanism for dynamic password generation. A Design of One-Time Password Mechanism Using Public Key Infrastructure by Hyun-Chul Kim et al [3] helped to explore various One-time Password algorithms and provided a comparison for the same. It also used a session identifier L and a random value R as inputs to a hash function to generate passwords. Balkis Hamdane et al [4] used the HMAC-Based One-Time Password Algorithm for TLS Authentication. They extended the TLS algorithm to include the HOTP algorithm. This TLS-HOTP protocol used symmetric keys, where client key is secured in a token. This provided a perfect base to develop a similar protocol for Wireless network security. One-time Passwords combined with a method to provide random seeding could be an improved solution.

The HMAC-based one time password (HOTP) algorithm [5]

The HOTP algorithm is based on an increasing counter value and a static symmetric key known only to the token and the validation service. In order to create the HOTP value, we will use the HMAC-SHA-1 algorithm. As the output of the HMAC-SHA-1 calculation is 160 bits, we must truncate this value to something that can be easily entered by a user.

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

Where:

- Truncate represents the function that converts an HMAC-SHA-1 value into an HOTP value
The Key (K), the Counter (C), and Data values are hashed high-order byte first.

The HOTP values generated by the HOTP generator are treated as big endian.

Managed Wi-Fi API [6]

This API is a .NET class library allowing you to control Wi-Fi (802.11) network adapters installed in your Windows machine programmatically. The library uses the Native Wi-Fi API. It acts as a wrapper around the Native Wi-Fi API making the functions easier to understand and use. The API contains methods to discover Wi-Fi networks, configure your wireless adapter, and connect to networks.

III. Analysis Of Existing Algorithm

With a strong foundation of the existing HOTP algorithm, it is necessary to know where the changes should be incorporated to adapt it to our specifications while retaining its strengths. The following flow diagram provides a graphical summary of the working and generation of OTP using this algorithm.

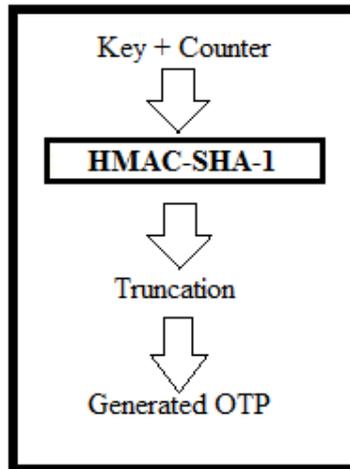


Fig. 1 Flow diagram of the existing algorithm

The heart of the algorithm is the HMAC-SHA-1 digest that is produced. HMAC is not a hash function. It is a message authentication code (MAC) that uses the hash function internally [6] The HMAC-SHA-1 uses a shared “Key” with the user and a “Counter” value as a seed to generate a 160 bits message digest. The counter value changes with every login and thus provides randomness. This 160 bits message digest is then supplied to a truncation function that generates a 6 digit OTP using modulo function. For better security it is recommended that we use 7-digit or 8-digit OTPs. Until recently, SHA1 was supposed to be practically impossible to break. The best attack possible was the birthday attack that required 2^{80} trials. In 2005, Professor Xiaoyun Wang and her associates in Tsinghua University and Shandong University of Technology had officially cracked the SHA-1 hashing algorithm [7] They were able to find a second pre-image that gave a collision in 2^{69} trials. A hash is considered broken if the collisions occur in less than that possible in a birthday attack. Thus the Chinese group proved that it was possible to break SHA1 hashes. Currently, the number of trials for collision has reduced to 2^{52} . Carefully engineered texts provide same hash for collision attacks. Also, there have been mentions of the dictionary attack possible on SHA1 hash. The Key used to generate the SHA1 message digest is used in plaintext. The counter value that is used to provide varying seeds is relatively small which leads to smaller repetition periods. Also the SHA1 hash produces a digest of 160 bits, which can be increased.

Although HMAC-SHA1 is not compromised due to the breaks in SHA1 hash function, it would be a good idea to start venturing for options and analyze them.

IV. Proposed Algorithm

Choosing a substitute hashing algorithm is one of the major decisions. A hash algorithm that retains the strengths of SHA1 and at the same time improves security without much overhead. Other alternatives comparable to SHA1 are;

TABLE I
Comparison of Various Hashing Algorithms [8]

Hashing Algorithm	Size of message digest	Time to hash 500 MB
SHA1	20 bytes	1644 ms
MD5	16 bytes	1462 ms
SHA256	32 bytes	5618 ms
SHA384	48 bytes	3839 ms
*SHA512	64 bytes	3820 ms
RIPEMD	20 bytes	7066 ms

- MD5 seems to be the fastest but, it has been broken so it should not be used.
- RIPEMD is the latest algorithm and is very secure. But, it is very slow, so it should be avoided.
- Among the other variations of SHA, SHA512 could be the best candidate. It has the largest digest size of 64 bytes and is also comparatively faster than the other two, SHA256 and SHA384.

Since this new algorithm can also derive from the idea of TSRG, a new module is introduced to secure the masterkey and generate counter values for the hash function randomly. This module will generate a random noise image that will be used for steganography.

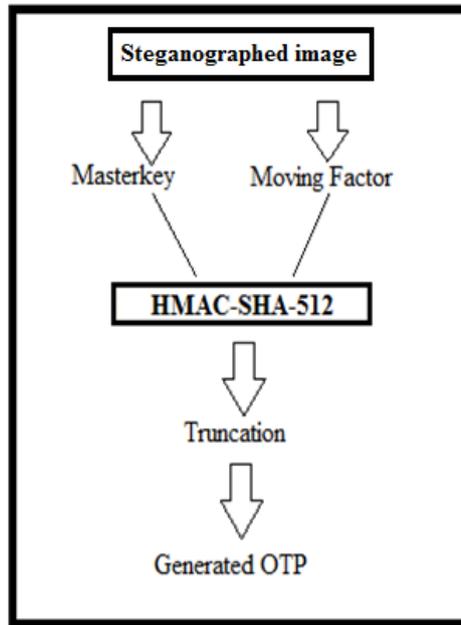


Fig. 2 Flow diagram of the proposed algorithm

The system consists of four modules, viz.

1. The user-interface.
2. The authentication server.
3. The masterkey generator.
4. The OTP generator.

The algorithm passes the following phases;

A. Phase 1: Registration

The server after successful validation of the user, requests the masterkey generator to generate a random masterkey which is steganographed in a random noise image. The random noise image perfectly blends into the noise on the network channel. Also simultaneously the first OTP is generated by the OTP generator. These user credentials are stored at the server and encrypted and sent to the user. The masterkey is stored on the user terminal and the OTP is displayed. The OTP does not persist on the user machine.

B. Phase 2: Search a network and request to connect

The user can search for existing Wi-Fi networks in the vicinity using the managed Wi-Fi API. When the user selects a network and tries to connect to it, the server authenticates him using the old OTP. The old OTP is then passed to the OTP generator. The random noise image contains pre-generated random pixels that are unused and unaltered by steganography. The OTP generator extracts consecutive RGB values for every attempt and appends the old OTP to generate the moving factor. This moving factor combined with the user masterkey is supplied to the HMAC-SHA-512 function which generates a digest. This digest on truncation gives the new OTP, which is sent to the user.

C. Phase 3: Connect to network and disconnect

The user receives the new OTP in a file on his device. The file automatically deletes after persisting for a specific interval and the OTP expires after a specific extended time. The user must retrieve this new OTP and use it before it expires. When the user is connected, the session starts and the database is updated accordingly. When the user disconnects or moves out of the network, the session ends.

D. Phase 4: Tracking usage and optional support

The user has an option to view and track his number of logins. When the OTP period is nearing closure, the user can request to generate a new masterkey steganographed image which will reset the OTP to new values.

V. Analysis Of Results

The proposed algorithm is still in the inception stage. It is mandatory to test it and understand its strengths and weaknesses. Its usability is based on three key concepts:

- 1) Time to generate OTPs – The overhead of generation of OTP should not over-shadow its security
- 2) Flexibility – The algorithm must provide, and accept varying levels of security depending on the masterkey and OTP size.
- 3) Scalability – The algorithm must adapt to different usage scenarios and load due to multiple access.

Specific tests to identify the following parameters were conducted and the results obtained from these testing were intuitive in identifying the best possible conditions under which the above mentioned key points could be exploited the most.

- 1) Time to generate image with different masterkey sizes.
- 2) Time to generate masterkey images of different sizes.
- 3) Time to generate a single OTP.
- 4) The period of repetition of OTPs.

A. Generate image with different masterkey sizes

The masterkey steganographed image of size 64 x 64 pixels was generated by varying the masterkey size. Greater the masterkey size means increased security from brute force attack. But, this comes with an overhead of time for generation. This data is represented in the graph below. As we can see, the time to generate the same image with different masterkey sizes does not increase proportionally. As the masterkey size increases, the image generation takes comparatively longer. This can be a concern when generating images of higher sizes, as they can accommodate longer masterkeys.

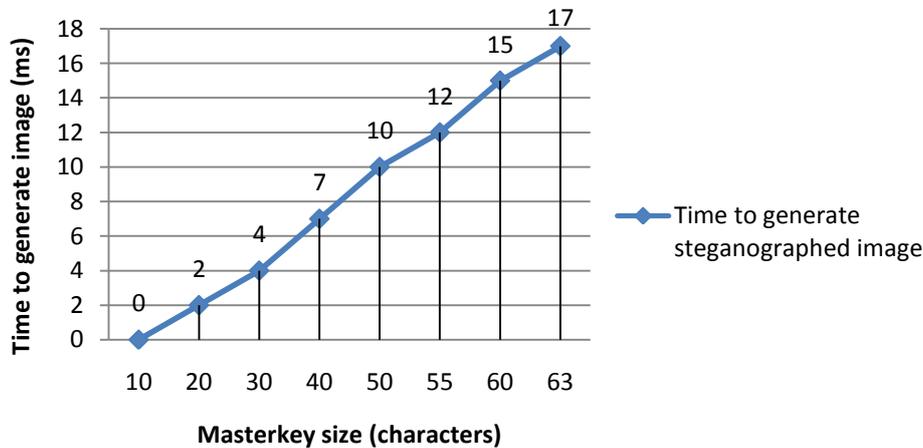


Fig. 3 Graph showing the time to generate a steganographed image while varying the masterkey size and keeping the resolution of the steganographed image constant

B. Generate masterkey images of different sizes

Varying the masterkey image size gives opportunity to generate more OTPs. It can provide more seeds for generation. But again this comes with an overhead of generation time. The above data is represented in the graph below. A major point to be noted is that for double the image size (from 64 to 128) the time for generation almost quadruples (from 19 msec to 73 msec). Hence, one must carefully consider before increasing the image size.

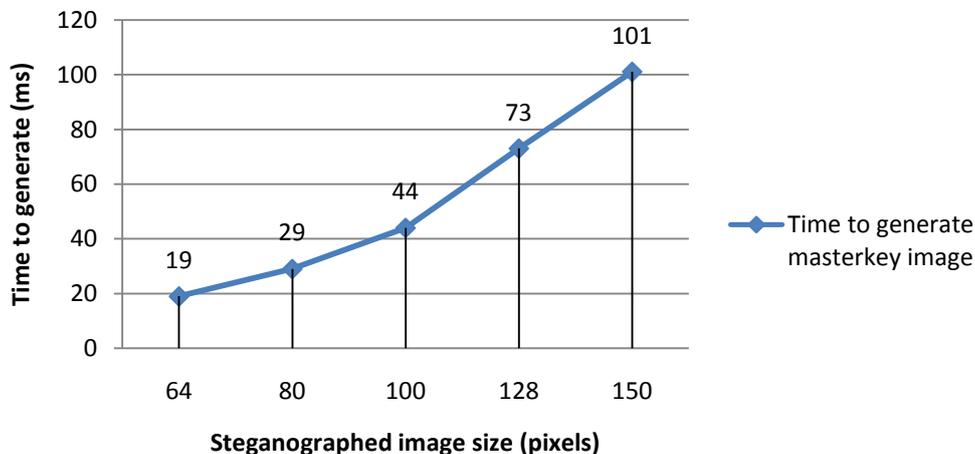


Fig. 4 Graph showing the time to generate a steganographed image while varying the resolution of the image and keeping the masterkey constant

C. Time to generate a single OTP

Since the seeds for generation of OTP are already generated in the masterkey image, the time to generate OTP is governed only by the algorithm to generate the digest. This time is almost constant over the length of OTP generated. The time for generation of OTP was found to range from 3 – 4 ms for OTP of size 6, 7, 8.

D. The period of repetition of OTPs

This is the most crucial parameter for evaluation. If the attacker knows the period of the OTP algorithm, he can monitor the user's usage and can reuse the OTP at the correct instant. Since the seeds for OTP are provided by the masterkey image, it decides the period of the OTP stream. Greater the size of the masterkey image, greater is the period of the OTP stream. For double the image size the stream period more than quadrupled. As is seen, for 64x64 it is 4021 values and for 128x128 it is 16237 values.

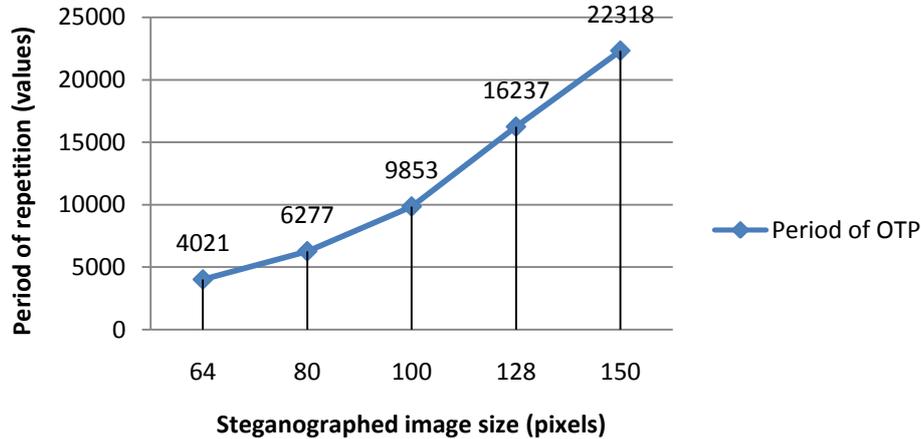


Fig. 5 Graph showing the variation of the repetition period of OTP depending on the resolution of the steganographed image

VI. Conclusion And Future Scope

A modified algorithm based on the HOTP algorithm was proposed, implemented and tested. The analysis of the results uncovered its strengths and weakness. The features of the original algorithm were imbibed with contributions from other OTP generation techniques. The algorithm still shows signs of large overheads with increased security. Various parameters such as the masterkey size, size of the noise image to be used for steganography, the OTP size, the window for deletion of the OTP file and the period for OTP repetition have to be standardized. Other issues that need to be addressed are of synchronization with the server and thrashing. Artificial intelligence can be used to study user behavior and adapt the algorithm accordingly. Hopefully the HMAC-SHA1 will suffice for a little more time, which will give us time to improve and optimize this algorithm for city-wide public Wi-Fi access.

References

- [1] Pablo Vidales, Alexander Manecke, Marcin SolarSKI, "Metropolitan Public Wi-Fi Access Based on Broadband Sharing"
- [2] Gamal Hussein, "Randomized Cryptosystems Attacks and Defenses"
- [3] Hyun-Chul Kim, Hong-Woo Lee, Kyung-Seok Lee, Moon-Seog Jun, "A Design of One – Time Password Mechanism using Public Key Infrastructure"
- [4] Balkis Hamdane, Ahmed Serhrouchni, Adrien Montfaucon, Sihem Guemara, "Using the HMAC-Based One-Time Password Algorithm for TLS Authentication"
- [5] RFC 4226: HOTP: An HMAC-Based One-Time Password Algorithm (Online), <http://tools.ietf.org/html/rfc4226#section-5>
- [6] Managed Wi-Fi API (Online), <http://managedWi-Fi.codeplex.com/>
- [7] SHA1 broken (Online), http://www.schneier.com/blog/archives/2005/02/sha1_broken.html
- [8] Hashing algorithms (Online), <http://stackoverflow.com/questions/800685/which-cryptographic-hash-function-should-i-choose>