International Journal of Scientific & Engineering Research Volume 8, Issue 10, October-2017
ISSN 2229-5518

Password Security Techniques

Rahul Chaudhary[1], Govind Prasad Arya[2]

Abstract— It is very difficult to secure passwords from hackers in this era because there are many tools present in the hacking world. In this paper I have discussed about the types of passwords and lengths of passwords chosen and the ones which should not be used, the type of passwords which are comparatively safer to use and the most common cracking techniques in use. The measures which can be applied in order to secure our passwords have also been described here. I have included algorithms which can be used to secure our passwords. The methods by which we can secure our passwords from attacks have also been described here. In this paper I have included the types of Wi-Fi passwords.

1 INTRODUCTION

A password is a set of characters used f–or user authentication to prove identity or access approval to gain access to a resource which is to be kept secret from those not allowed access. We can't store password in plain text because of cyber threads. To secure our password from various cyber threads we use many methods and concepts like algorithms
Hashing is a type of algorithm which takes any size of data and turns it into a fixed-length of data.
Modern Hashing Algorithm:

- MD-5
- SHA-1
- SHA-2
- SHA-3

Hashing Passwords algorithms
There are currently three algorithms which are safe to use:

- PBKDF2
- bcrypt
- scrypt

As we all known in today world hacking is major issue. There are many attacks which takes our password without our permission which is dangerous because while doing so our personal data is in unauthorized person.
There are many threads:

- Dictionary attacks
- Brute-force attacks
- Rainbow attacks
- *Denial-of-Service Attack*
- Man-in-the-Middle Attack
- Birthday attacks

I have discussed in this paper how prevent our passwords from above attacks. As password is most important part of our life so we need to make it as a secure password.

2. TYPES OF PASSWORDS

2.1. Types of computer passwords include

Power-on password:-This password prevents your system from being powered on by unauthorized users. Hard drive password.There are two kinds of hard:
-userhard drive password and
- master hard drive password for administrators
Supervisor (BIOS) password
The BIOS or Supervisor password protects the system information stored in the BIOS. A password is needed for the user to access the BIOS Setup Utility to change system configurations.
Operation System password
Operating systems include for instance, Windows, Windows 7, Windows 8, Mac and Linux.

2.2. The WEP, WPA, and WPA2 Wi-Fi Passwords

2.2.1 Wired Equivalent Privacy (WEP)

Wired Equivalent Privacy (WEP) is the most widely used Wi-Fi security algorithm in the world. This is a function of age, backwards compatibility, and the fact that it appears first in the encryption type selection menus in many router control panels.
WEP was ratified as a Wi-Fi security standard in September of 1999. The first versions of WEP weren't particularly strong, even for the time they were released, because U.S. restrictions on the export of various cryptographic technologies led to manufacturers restricting their devices to only 64-bit encryption. When the restrictions were lifted, it was increased to 128-bit. Despite the introduction of 256-bit WEP encryption, 128-bit remains one of the most common implementations.Despite revisions to the algorithm and an increased key size, over time numerous security flaws were discovered in the WEP standard and, as computing power increased, it became easier and easier to exploit them. As early as 2001 proof-of-concept exploits were floating around and by 2005 the FBI gave a public demonstration (in an effort to increase awareness of WEP's weaknesses) where they cracked WEP passwords in minutes using freely available software. Despite various improvements, workaround, and other attempts to shore up the WEP system, it remains highly vulnerable and systems that rely on WEP should be upgraded or, if security upgrades are not an option, replaced. The Wi-Fi Alliance officially retired WEP in 2004.

2.2.2 Wi-Fi Protected Access (WPA)

Wi-Fi Protected Access was the Wi-Fi Alliance's direct response and replacement to the increasingly apparent vulnerabilities of the WEP standard. It was formally adopted in 2003, a year before WEP was officially re-

tired. The most common WPA configuration is WPA-PSK (Pre-Shared Key). The keys used by WPA are 256-bit, a significant increase over the 64-bit and 128-bit keys used in the WEP system. Some of the significant changes implemented with WPA included message integrity checks (to determine if an attacker had captured or altered packets passed between the access point and client) and the Temporal Key Integrity Protocol (TKIP). TKIP employs a per-packet key system that was radically more secure than fixed key used in the WEP system. TKIP was later superseded by Advanced Encryption Standard (AES). Despite what a significant improvement WPA was over WEP, the ghost of WEP haunted WPA. TKIP, a core component of WPA, was designed to be easily rolled out via firmware upgrades onto existing WEP-enabled devices. As such it had to recycle certain elements used in the WEP system which, ultimately, were also exploited. WPA, like its predecessor WEP, has been shown via both proof-of-concept and applied public demonstrations to be vulnerable to intrusion. Interestingly the process by which WPA is usually breached is not a direct attack on the WPA algorithm (although such attacks have been successfully demonstrated) but by attacks on a supplementary system that was rolled out with WPA, Wi-Fi Protected Setup (WPS), designed to make it easy to link devices to modern access points.

2.2.3 Wi-Fi Protected Access II (WPA2)

WPA has, as of 2006, been officially superseded by WPA2. One of the most significant changes between WPA and WPA2 was the mandatory use of AES algorithms and the introduction of CCMP (Counter Cipher Mode with Block Chaining Message Authentication Code Protocol) as a replacement for TKIP (still preserved in WPA2 as a fallback system and for interoperability with WPA).

Currently, the primary security vulnerability to the actual WPA2 system is an obscure one (and requires the attacker to already have access to the secured Wi-Fi network in order to gain access to certain keys and then perpetuate an attack against other devices on the network). As such, the security implications of the known WPA2 vulnerabilities are limited almost entirely to enterprise level networks and deserve little to no practical consideration in regard to home network security.

Unfortunately, the same vulnerability that is the biggest hole in the WPA armor, the attack vector through the Wi-Fi Protected Setup (WPS), remains in modern WPA2-capable access points. Although breaking into a WPA/WPA2 secured network using this vulnerability requires anywhere from 2-14 hours of sustained effort with a modern computer, it is still a legitimate security concern and WPS should be disabled (and, if possible, the firmware of the access point should be flashed to a distribution that doesn't even support WPS so the attack vector is entirely removed).

3 ALTERNATIVES TO PASSWORDS FOR AUTHENTICATION

- Single-use passwords. Having passwords which are only valid once makes many potential attacks ineffective. Most users find single use passwords extremely inconvenient. They have, however, been widely implemented in personal online banking, where they are known as Transaction Authentication Numbers (TANs). As most home users only perform a small number of transactions each week, the single use issue has not led to intolerable customer dissatisfaction in this case.

- Time-synchronized one-time passwords are similar in some ways to single-use passwords, but the value to be entered is displayed on a small (generally pocketable) item and changes every minute or so.
- PassWindow one-time passwords are used as single-use passwords, but the dynamic characters to be entered are visible only when a user superimposes a unique printed visual key over a server generated challenge image shown on the user's screen.
- Access controls based on public key cryptography e.g. ssh. The necessary keys are usually too large to memorize (but see proposal Passmaze) and must be stored on a local computer, security token or portable memory device, such as a USB flash drive or even floppy disk.
- Biometric methods promise authentication based on unalterable personal characteristics, but currently (2008) have high error rates and require additional hardware to scan, for example, fingerprints, irises, etc. They have proven easy to spoof in some famous incidents testing commercially available systems, for example, the gummie fingerprint spoof demonstration, and, because these characteristics are unalterable, they cannot be changed if compromised; this is a highly important consideration in access control as a compromised access token is necessarily insecure.
- Single sign-on technology is claimed to eliminate the need for having multiple passwords. Such schemes do not relieve user and administrators from choosing reasonable single passwords, nor system designers or administrators from ensuring that private access control information passed among systems enabling single sign-on is secure against attack. As yet, no satisfactory standard has been developed.
- Evaluating technology is a password-free way to secure data on removable storage devices such as USB flash drives. Instead of user passwords, access control is based on the user's access to a network resource.
- Non-text-based passwords, such as graphical passwords or mouse-movement based passwords. Graphical passwords are an alternative means of authentication for log-in intended to be used in place of conventional password;

they use images, graphics or colors instead of letters, digits or special characters. One system requires users to select a series of-faces as a password, utilizing the human brain's ability to recall faces easily. In some implementations the user is required to pick from a series of images in the correct sequence in order to gain access. Another graphical password solution creates a one-time password using a randomly generated grid of images. Each time the user is required to authenticate, they look for the images that fit their pre-chosen categories and enter the randomly generated alphanumeric character that appears in the image to form the one-time password. So far, graphical passwords are promising, but are not widely used. Studies on this subject have been made to determine its usability in the real world. While some believe that graphical passwords would be harder to crack, others suggest that people will be just as likely to pick common images or sequences as they are to pick common passwords.

- 2D Key (2-Dimensional Key) is a 2D matrix-like key input method having the key styles of multiline passphrase, crossword, ASCII/Unicode art, with optional textual semantic noises, to create big password/key beyond 128 bits to realize the MePKC (Memorizable Public-Key Cryptography) using fully memorizable private key upon the current private key management technologies like encrypted private key, split private key, and roaming private key. Cognitive passwords use question and answer cue/response pairs to verify identity.

.

3.1. Password Cracking Techniques

3.1.1 Dictionary attacks

Dictionary attacks quickly compare a set of known dictionary-type words.including many common passwords against a password database. This database is a text file with hundreds if not thousands of dictionary words typically listed in alphabetical order.

3.1.2 Brute-force attacks

Brute-force attacks try every combination of numbers, letters, and special characters until the password is discovered. Many password-cracking utilities let you specify such testing criteria as the character sets, password length to try, and known characters (for a "mask" attack).

3.1.3 Rainbow attacks

A rainbow password attack uses rainbow cracking to crack various password hashes for LM, NTLM, Cisco PIX, and MD5 much more quickly

Fig 1.0

Scheme	Stolen Password	Stolen Data	Stolen Data and Password
Password Safe	N/A	74.6 secs	$\leq$ 74.6 secs
LPWA	0.5 secs	N/A	N/A
PwdHash	0.1 secs	0.1 secs	0.1 secs
Our Scheme	116 days	116 days	2.8 hours

Table 3: Resistance to a dictionary attack under three attack scenarios—Times to test 100,000 password guesses using a fast modern PC.

and with extremely high success rates (near 100 percent). Password-cracking speed is increased in a rainbow attack because the hashes are precalculated and thus don't have to be generated individually on the fly as they are with dictionary and brute-force cracking methods.

There's a length limitation because it takes significant time to generate these rainbow tables. Given enough time, a sufficient number of tables will be created. Of course, by then, computers and applications likely have different authentication mechanisms and hashing standards — including a new set of vulnerabilities — to contend with.

3.2. Password-Based Attacks

A common denominator of most operating system and network security plans is password-based access control. This means your access rights to a computer and network resources are determined by who you are, that is, your user name and your password.

Older applications do not always protect identity information as it is passed through the network for validation. This might allow an eavesdropper to gain access to the network by posing as a valid user.

When an attacker finds a valid user account, the attacker has the same rights as the real user. Therefore, if the user has administrator-level rights, the attacker also can create accounts for subsequent access at a later time. After gaining access to your network with a valid account, an attacker can do any of the following:

Obtain lists of valid user and computer names and network information.

Modify server and network configurations, including access controls and routing tables.

Modify, reroute, or delete your data.

3.3 Denial-of-Service Attack

Unlike a password-based attack, the denial-of-service attack prevents normal use of your computer or network by valid users.After gaining access to your network, the attacker can do any of the following:
Randomize the attention of your internal Information Systems staff so that they do not see the intrusion immediately, which allows the attacker to make more attacks during the diversion. Send invalid data to applications or network services, which causes abnormal termination or behavior of the applications or services.Flood a computer or the entire network with traffic until a shutdown occurs because of the overload. Block traffic, which results in a loss of access to network resources by authorized users.

3.4 Man-in-the-Middle Attack

As the name indicates, a man-in-the-middle attack occurs when someone between you and the person with whom you are communicating is actively monitoring, capturing, and controlling your communication transparently. For example, the attacker can re-route a data exchange. When computers are communicating at low levels of the network layer, the computers might not be able to determine with whom they are exchanging data. Man-in-the-middle attacks are like someone assuming your identity in order to read your message. The person on the other end might believe it is you because the attacker might be actively replying *as you* to keep the exchange going and gain more information. This attack is capable of the same damage as an application-layer attack, described later in this section.

3.5 Birthday attacks

This attack exploits the Birthday paradox , which in brief states that, having a large set of user password digests, the probability of generating a password which digest collides with at least one of the digests in the set is very much higher than what you would intuitively expect. And this probability increases dramatically as the size of the set (the number of users) augments.

4. ALGORITHMS

4.1 Hashing

Hashing is a type of algorithm which takes any size of data and turns it into a fixed-length of data. This is often used to ease the retrieval of data as you can shorten large amounts of data to a shorter string (which is easier to compare). For instance let's say you have a DNA sample of a person, this would consist of a large amount of data (about 2.2 – 3.5 MB), and you would like to find out to who this DNA sample belongs to. You could take all samples and compare 2.2 MB of data to all DNA samples in the database, but comparing 2.2 MB against 2.2 MB of data can't take quite a while, especially when you need to traverse thousands of samples. This is where hashing can come in handy, instead of comparing the data, you calculate the hash of this data (in reality, several hashes will be calculated for the different locations on the chromosomes, but for the sake of the example let's assume it's one hash), which will return a fixed length value of, for instance, 128 bits. It will be easier and faster to query a database for 128-bits than for 2.2 MB of data.
The main difference between hashing and encryption is that a hash is not reversible. When we are talking about cryptographic hash functions, we are referring to hash functions which have these properties:

- It is easy to compute the hash value for any given message.
- It is infeasible to generate a message that has a given hash.
- It is infeasible to modify a message without changing the hash.
- It is infeasible to find two different messages with the same hash.

The hash function should be resistant against these properties:

- Collisions (two different messages generating the same hash)
- Pre-image resistance: Given a hash h it should be difficult to find any message m such that h = hash(m).
- Resistance to second-preimages: given m, it is infeasible to find m' distinct from m and such that MD-5(m) = MD-5(m').

5. Modern Hashing Algorithms

Some hashing algorithms you may encounter are:

- MD-5
- SHA-1
- SHA-2
- SHA-3

5.1 MD-5

MD-5 is a hashing algorithm which is still widely used but cryptographically flawed as it's prone to collisions. MD-5 is broken in regard to collisions, but not in regard of preimages or second-preimages. The first attacks on MD-5 were published in 1996, this was in fact an attack on the compression of MD-5 rather than MD-5 itself. In 2004 a theoretical attack was produced which allowed for weakening the pre-image resistance property of MD-5. In practice the attack is way too slow to be useful.

5.2 SHA

SHA or Secure Hashing Algorithm is a family of cryptographic hash functions published by the National Institute of Standards and Technology (NIST) as a U.S.

Federal Information Processing Standard (FIPS). Currently three algorithms are defined:

- SHA-1: A 160-bit hash function which resembles the earlier MD-5 algorithm. This was designed by the National Security Agency (NSA) to be part of the Digital Signature Algorithm. Cryptographic weaknesses were discovered in SHA-1, and the standard was no longer approved for most cryptographic uses after 2010.
- SHA-2: A family of two similar hash functions, with different block sizes, known as SHA-256 and SHA-512. They differ in the word size; SHA-256 uses 32-bit words where SHA-512 uses 64-bit words. There are also truncated versions of each standardized, known as SHA-224 and SHA-384. These were also designed by the NSA.
- SHA-3: SHA-3 is not yet defined. NIST is working on the exact parameters they will use; SHA-3 will beKeccak, or "close enough", but not necessarily the Keccak which was submitted (it is a configurable function, and they seem to want to tweak the parameters a bit differently than what was first proposed).

Note that while SHA-1 is "cryptographically broken" the properties we seek in a password hashing algorithm are still valid. In the real world finding a password hashing algorithm built on SHA-1 is still secure in the sense, that if it's implemented there is no reason to assume it should be immediately changed to something newer.

6 HASHING PASSWORDS ALGORITHMS

There are currently three algorithms which are safe to use:

- PBKDF2
- bcrypt
- scrypt
- PBKDF2

6.1 PBKDF2 is an algorithm which is used to derive keys. It wasn't intended for password hashing, but due to its property of being slow, it lends itself quite well for this purpose. The resulting derived key (HMAC) can actually be used to securely store passwords. It's not the ideal function for password hashing, but it's easy to implement and it's built upon SHA-1 or SHA-2 hashing algorithms (any HMAC will do, but these are the most common used ones and the most secure ones). Wait, didn't you say SHA-1 and SHA-2 were bad to use when hashing passwords? Yes indeed, that's why we use PBKDF2 to make the hashing a lot slower. You still will need to choose your hashing algorithm carefully, PBKDF2+Keccak is a substantially worse choice than PBKDF2+SHA-256, which is already somewhat worse than PBKDF2+SHA-512 if your server is a 64-bit PC.

To derive a key PBKDF2 does the following:

DK = PBKDF2(PRF, Password, Salt, c, dkLen)

Where DK is the derived key, PRF is the preferred HMAC function (this can be a SHA-1/2 HMAC, the password is used as a key for the HMAC and the salt as text), c is the amount of iterations and dkLen is the length of the derived key. A salt should, by definition of the standard, be at least 64-bits of length and the minimum amount of iterations should be 1024. What the algorithm will do is SHA-1-HMAC(password+salt), and repeat the calculation 1024 times on the result. This means the hashing of a password will be 1024 times slower. Still this does not actually offer a lot of protection when brute-forcing on distributed systems or GPU (Graphic Processing Unit).

There's also a caveat when the password exceeds 64 bytes, the password will be shortened by applying a hash to it by the PBKDF2 algorithm so it does not exceed the block size. For instance when using HMAC-SHA-1 a password longer than 64 bytes will be reduced to SHA-1(password), which is 20 bytes in size. This means passwords longer than 64 bytes do not provide additional security when it comes to breaking the key used to make the HMAC, but may even reduce security as the length of the key will be reduced (note that even when reduced to 20 bytes, currently our great-great-great-great-great-great-great-great-great-great grandchildren will be long dead before the key is brute forced).

6.2 bcrypt

bcrypt is currently the defacto secure standard for password hashing. It's derived from the Blowfish block cipher which, to generate the hash, uses look up tables which are initiated in memory. This means a certain amount of memory space needs to be used before a hash can be generated. This can be done on CPU, but when using the power of GPU it will become a lot more cumbersome due to memory restrictions. Bcrypt has been around for 14 years, based on a cipher which has been around for over 20 years. It's been well vetted and tested and hence considered the standard for password hashing.

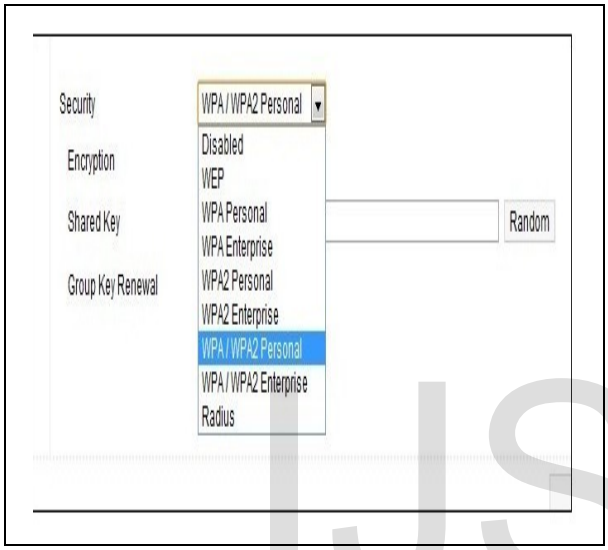
There is actually one weakness, FPGA processing units. When bcrypt was originally developed its main threat was custom ASICs specifically built to attack hash functions. These days those ASICs would be GPUs (password brute-forcing can actually still run on GPU, but not in full parallelism) which are cheap to purchase and are ideal for multithreaded processes such as password brute-forcing.

FPGAs (Field Programmable Gate Arrays) are similar to GPUs but the memory management is very different. On these chips brute-forcing bcrypt can be done more efficiently than on GPUs, but if you have a long enough passwords it will still be unfeasible.

6.3 Scrypt

For password hashing, the current fashion is to move the problem away to another level; instead of doing a lot of hash function invocations, concentrate on an operation which is hard for anything else than a PC, e.g. random memory accesses. That's what scrypt is about. Scrypt is another hashing algorithm which has the same proper-

ties as bcrypt, except that when you increase rounds, it exponentially increases calculation time and memory space required to generate the hash. Scrypt was created as response to evolving attacks on bcrypt and is completely unfeasible when using FPGAs or GPUs due to memory constraints. Scrypt requires the storage of a series of intermediate state data "snapshots", which are used in further derivation operations. These snapshots, stored in memory, grow exponentially compared when rounds increase. So adding a round, will make it exponentially harder to brute force the password. Scrypt is still relatively new compared to bcrypt and has only been around for a couple of years, which makes it less vetted than bcrypt.



7. Password Strength

Strong passwords

Apart from choosing a good hashing algorithm you should also force your users to choose a password which is built up of at least eight, random characters. Unfortunately people aren't designed to remember and generate random sequences of characters. This is why we force our users to make passwords which contain numbers, letters, signs and at least one capital letter. But how does this help in regard to password hashing?

To attack hashed passwords there are different strategies:

- Dictionary Attacks
- Brute-force
- Rainbow Tables (generate everything upfront in a database and do a look up for each hash)

With a dictionary attack you will try to use word lists, these can consist of mostly used passwords, words, names, years, etc. For each word you will run the hashing algorithm and see if the generated hash is the same as the hash in the database. If this is the case then you know that the word from which you derived the hash is the password.With a brute-force attack you will try all possible combinations of characters. When using passwords of at least eight characters long, only using the ASCII characters set, there are 128^8 possibilities of passwords.

To show the importance of the length of a password:

These days, using a single, modern GPU, you can process about 10.323.000.000 passwords per *second* when brute-forcing plain MD-5. With this speed, when using a password of eight random characters, it will take about eighty days to generate every single possibility. This single GPU only cost about 500 USD (AMD Radeon 6990).People have actually constructed clusters which contain 25 of these cards, optimized it and managed to generate 350 *billion* passwords per second. This means they can generate all possible passwords of eight random characters long in less than two days.

Now when you add one character to the password, the possibilities will be 128^9. With previous calculation of 350 billion it will now take 305 days. 10 characters > 106 years. This seems long, but we need to take into account Moore's law:

Moore's law is the observation that, over the history of computing hardware, the number of transistors on integrated circuits doubles approximately every two years. The period often quoted as "18 months" is due to Intel executive David House, who predicted that period for a doubling in chip performance (being a combination of the effect of more transistors and their being faster).

Computers have become faster and faster over the years, which is something we need to take into account. From a crypto graphical point of view, 106 years is still a short period. We want infinity (something which will take several hundred-thousand to millions of years).

7.1. Prevention from attacks

- Brute-force

By iterating the hash function to a number like 1,000 (minimum recommended), the overhead of password digest creation for the user at sign-up or sign-in time is not significant, but the accumulated cost for a brute force attacker generating millions of digests will be very considerable. Remember that one of the best ways to protect your encrypted data is making the cost of breaking your security too high to be worth the effort.

- Dictionary Attacks

By adding a random salt, the weakness of the dictionary-based passwords many people use is reduced (they are no longer dictionary words), and the possibility of the digest appearing on a set of digests previously created by the attacker is minimal.

- Birthday Attacks

By adding a random salt the possibilities of a birthday attack to succeed are minimum, because the attacker would have to attack each password separately, and not the set of passwords as a whole, to find a collision. This is because he/she would have to find a password that creates the same digest as the attacked one using the same salt which was used for digesting it, which is different for each password (this is, it would become a brute force attack).

• Unlike dictionary and brute-force attacks, rainbow attacks cannot be used to crack password hashes of unlimited length. The current maximum length for Microsoft LM hashes is 14 characters, and the maximum is up to 16 characters (dictionary-based) for Windows Vista and 7 hashes.

7.2. Securing your accounts

7.2.1 Make your password a sentence:

A strong password is a sentence that is at least 12 characters long. Focus on positive sentences or phrases that you like to think about and are easy to remember (for example, "I love country music."). On many sites, you can even use spaces!

7.2.2 Unique account, unique password:

Having separate passwords for every account helps to thwart cybercriminals. At a minimum, separate your work and personal accounts and make sure that your critical accounts have the strongest passwords.

7.2.3 Write it down and keep it safe:

Everyone can forget a password. Keep a list that's stored in a safe, secure place away from your computer. You can alternatively use a service like a password manager to keep track of your passwords.

7.2.4 Lock down your login:

Fortify your online accounts by enabling the strongest authentication tools available, such as biometrics, security keys or a unique one-time code through an app on your mobile device. Your usernames and passwords are not enough to protect key accounts like email, banking and social media.

7.3 Don't use as a password

- using a 'standard' word such as boss, master, doall, passwd
- using a dictionary word or the name of the business
- Using repeating letters or numerals (AAAAAA, 111111 and so on).
- Name of parent.
- Update Your Router and Upgrade to Third Party Firmware If Possible.
- Change Your Router's Password
- Name of best friends.
- Common name

8. CONCLUSION

Today many users (including those who should know better) fail to take secure steps to protect their passwords. Often the cause is not a failure to understand that strong passwords are important, but rather frustration with the difficulty of doing the right thing. In my study I informed you to make strong password and to protect yourself from attacks.

9 ACKNOWLEDGMENTS

This material is based upon Internet resources.

10 References

[1]. Jasypt.java simplified encryption
[2]. http://www.howtogeek.com/68403/how-to-secure-your-wi-fi-network-against-intrusion/
[3]. StaySafeOnline.org
[4]. password-wiki
[5]. http://www.passport.net.
[6].A Convenient Method for Securely Managing Passwords – Research paper by J. Alex Halderman, Brent Waters, Edward W. Felten
[7]. IT security community blog
[8]. Articsoft
[9]. http://xkcd.com/936/
[10]. http://www.passwordresearch.com