Republic of Yemen
University of Aden
Faculty of Computer and
Information Technology

Hash Fanction

Prepared by:

Sarah Mohammed Ahmed No.B181904018

Heba Mohammed Ali No.B181903028

Supervised By:

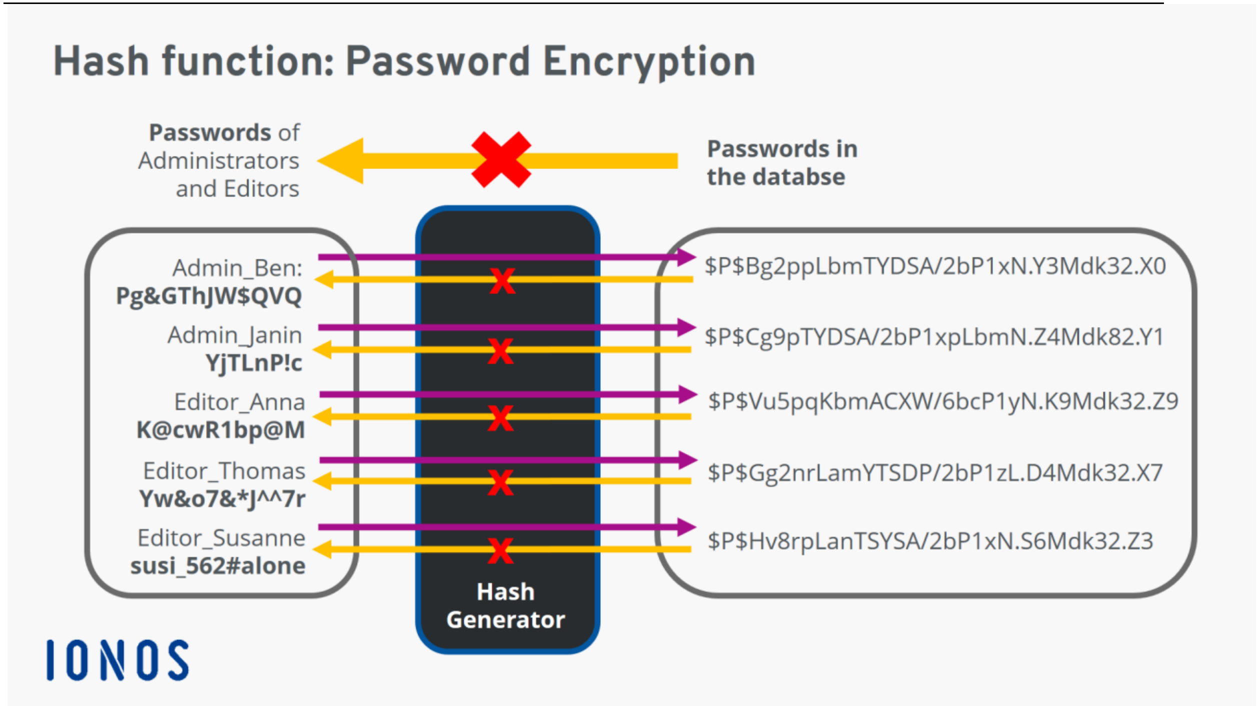
Eng. Aryam Essam

Republic of Yemen
University of Aden
Faculty of Computer and
Information Technology

hash function

Definition

A **hash function** converts strings of different length into fixed-length strings known as hash values or digests. You can use hashing to scramble passwords into strings of authorized characters for example. The output values cannot be inverted to produce the original input


Hash function: Password Encryption
Passwords of Administrators and Editors
Passwords in the databse
Admin_Ben:
Pg&GThJW$QVQ
Admin_Janin
YjTLnP!c
Editor_Anna
K@cwR1bp@M
Editor_Thomas
Yw&o7&*J^^7r
Editor_Susanne
susi_562#alone
PBg2ppLbmTYDSA/2bP1xN.Y3Mdk32.X0
PCg9pTYDSA/2bP1xpLbmN.Z4Mdk82.Y1
PVu5pqKbmACXW/6bcP1yN.K9Mdk32.Z9
PGg2nrLamYTSDP/2bP1zL.D4Mdk32.X7
PHv8rpLanTSYSA/2bP1xN.S6Mdk32.Z3
Hash Generator
IONOS

Types of Hash functions

There are many hash functions that use numeric or alphanumeric keys. This article focuses on discussing different hash functions:

1. **Division Method.**
2. **Mid Square Method.**
3. **Folding Method.**
4. **Multiplication Method.**

Let’s begin discussing these methods in detail.

1. Division Method:

This is the most simple and easiest method to generate a hash value. The hash function divides the value k by M and then uses the remainder obtained.

Formula:

h(K) = k mod M
Here,
***k** is the key value, and*
***M** is the size of the hash table.*

It is best suited that **M** is a prime number as that can make sure the keys are more uniformly distributed. The hash function is dependent upon the remainder of a division.

Example:
k = 12345
M = 95
h(12345) = 12345 mod 95
= 90

k = 1276
M = 11
h(1276) = 1276 mod 11
=

Pros:

1. This method is quite good for any value of M.
2. The division method is very fast since it requires only a single division operation.

Cons:

1. This method leads to poor performance since consecutive keys map to consecutive hash values in the hash table.
2. Sometimes extra care should be taken to chose value of M.

2.Mid Square Method:

The mid square method is a very good hashing method. It involves two steps to compute the hash value-

1. Square the value of the key k i.e. k^2
2. Extract the middle **r** digits as the hash value.

Formula:
h(K) = h(k x k)

Here,
***k** is the key value.*

The value of **r** can be decided based on the size of the table.
Example:
Suppose the hash table has 100 memory locations. So r = 2 because two digits are required to map the key to the memory location.

k = 60
k x k = 60 x 60
= 3600
h(60) = 60

The hash value obtained is 60

Pros:

1. The performance of this method is good as most or all digits of the key value contribute to the result. This is because all digits in the key contribute to generating the middle digits of the squared result.
2. The result is not dominated by the distribution of the top digit or bottom digit of the original key value.

Cons:

1. The size of the key is one of the limitations of this method, as the key is of big size then its square will double the number of digits.
2. Another disadvantage is that there will be collisions but we can try to reduce collisions.

3. Digit Folding Method:

This method involves two steps:

1. Divide the key-value **k** into a number of parts i.e. **k1, k2, k3,….,kn**, where each part has the same number of digits

except for the last part that can have lesser digits than the other parts.

2. Add the individual parts. The hash value is obtained by ignoring the last carry if any.

Formula:

k = k1, k2, k3, k4, ….., kn

s = k1+ k2 + k3 + k4 +….+ kn

h(K)= s

Here,

s** is obtained by adding the parts of the key **k

Example:

k = 12345

k1 = 12, k2 = 34, k3 = 5

s = k1 + k2 + k3

= 12 + 34 + 5

= 51

h(K) = 51

Note:

The number of digits in each part varies depending upon the size of the hash table. Suppose for example the size of the hash table is 100, then each part must have two digits except for the last part that can have a lesser number of digits.

4.Multiplication Method

This method involves the following steps:

1. Choose a constant value A such that 0 < A < 1.
2. Multiply the key value with A.
3. Extract the fractional part of kA.
4. Multiply the result of the above step by the size of the hash table i.e. M.
5. The resulting hash value is obtained by taking the floor of the result obtained in step 4.

Formula:

h(K) = floor (M (kA mod 1))
Here,
***M** is the size of the hash table.*
***k** is the key value.*
***A** is a constant value.*

Example:
k = 12345
A = 0.357840
M = 100

*h(12345) = floor[100 (12345*0.357840 mod 1)]*
= floor[100 (4417.5348 mod 1)]
= floor[100 (0.5348)]
= floor[53.48]
= 53

Pros:
The advantage of the multiplication method is that it can work with any value of between 0 and 1, although there are some values that tend to give better results than the rest.

Cons:
The multiplication method is generally suitable when the table size is the power of two, then the whole process of computing the index by the key using multiplication hashing is very fast.

The properties of hash functions:

Hash functions are designed so that they have the following **properties**:

One-way

Once a hash value has been generated, it must be **impossibleto convert it back** into the original data. For instance, in the example above, there must be no way of converting “$P$Hv8rpLanTSYSA/2bP1xN.S6Mdk32.Z3” back into “susi_562#alone”.

Collision-free

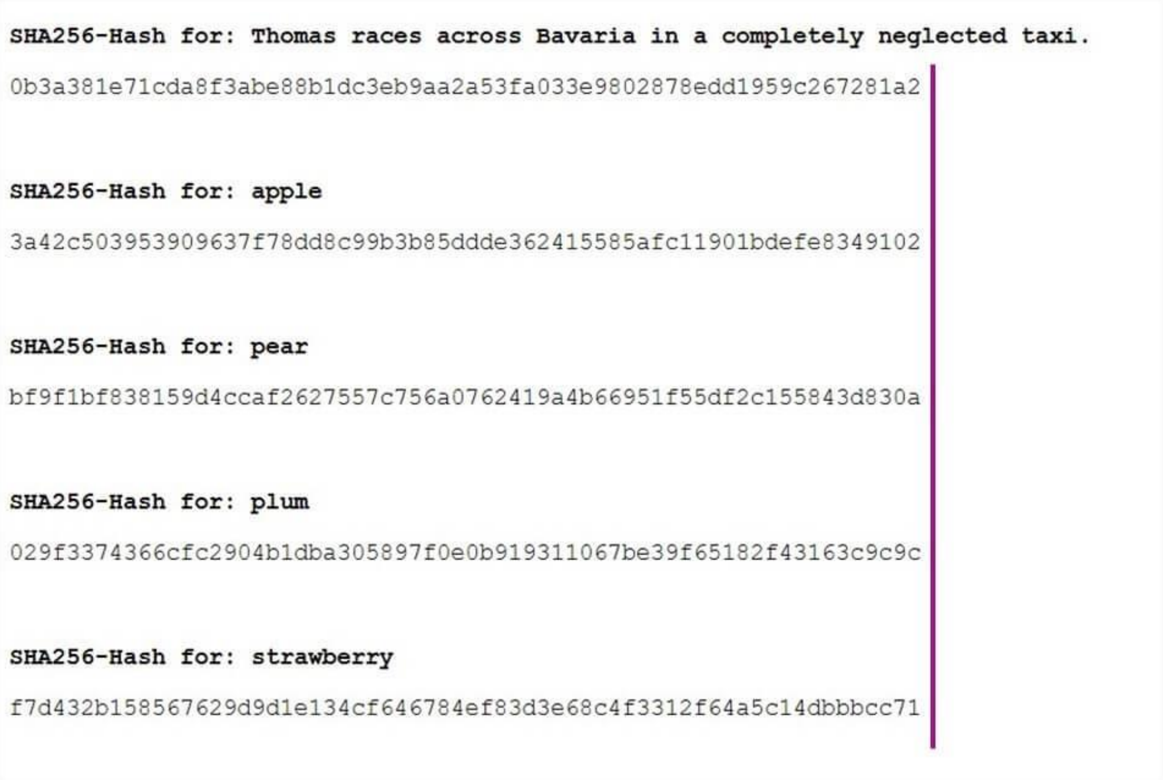
For a hash function to be collision-free, no two strings can map to the same output hash. In other words, every input string must generate a unique output string. This type of hash function is also referred to as a **cryptographic hash function**. In the example hash function above, there are no identical hash values, so there are **no “collisions”** between the output strings. Programmers use advanced technologies to prevent such collisions.

Lightning-fast

If it takes too long for a hash function to compute hash values, the procedure is not much use. Hash functions must, therefore, be very **fast**. In databases, hash values are stored in so-called hash tables to ensure fast access.

A hash value:

A hash value is the output string generated by a hash function. No matter the input, all of the output strings generated by a particular hash function are of the **same length**. The length is defined by the type of hashing technology used. The output strings are created from a **set of authorized characters** defined in the hash function.



```
SHA256-Hash for: Thomas races across Bavaria in a completely neglected taxi.

0b3a381e71cda8f3abe88b1dc3eb9aa2a53fa033e9802878edd1959c267281a2


SHA256-Hash for: apple

3a42c503953909637f78dd8c99b3b85ddde362415585afc11901bdefe8349102


SHA256-Hash for: pear

bf9f1bf838159d4ccaf2627557c756a0762419a4b66951f55df2c155843d830a


SHA256-Hash for: plum

029f3374366cfc2904b1dba305897f0e0b919311067be39f65182f43163c9c9c


SHA256-Hash for: strawberry

f7d432b158567629d9d1e134cf646784ef83d3e68c4f3312f64a5c14dbbbcc71
```

Hash values generated using the SHA256 function are always of the same length, irrespective of the number and type of characters in the input string.

The hash value is the result calculated by the hash function and algorithm. Because hash values are unique, like human fingerprints, they are also referred to as “**fingerprints**”. If you take the lower-case letters “a” to “f” and the digits “0” to “9” and define a hash value length of 64 characters, there are **1.1579209e+77 possible output values** – that’s 70 followed by 24 zeros! This shows that even with shorter strings, you can still generate acceptable fingerprints.

The hash values in the example above can be generated with just a few lines of PHP code:

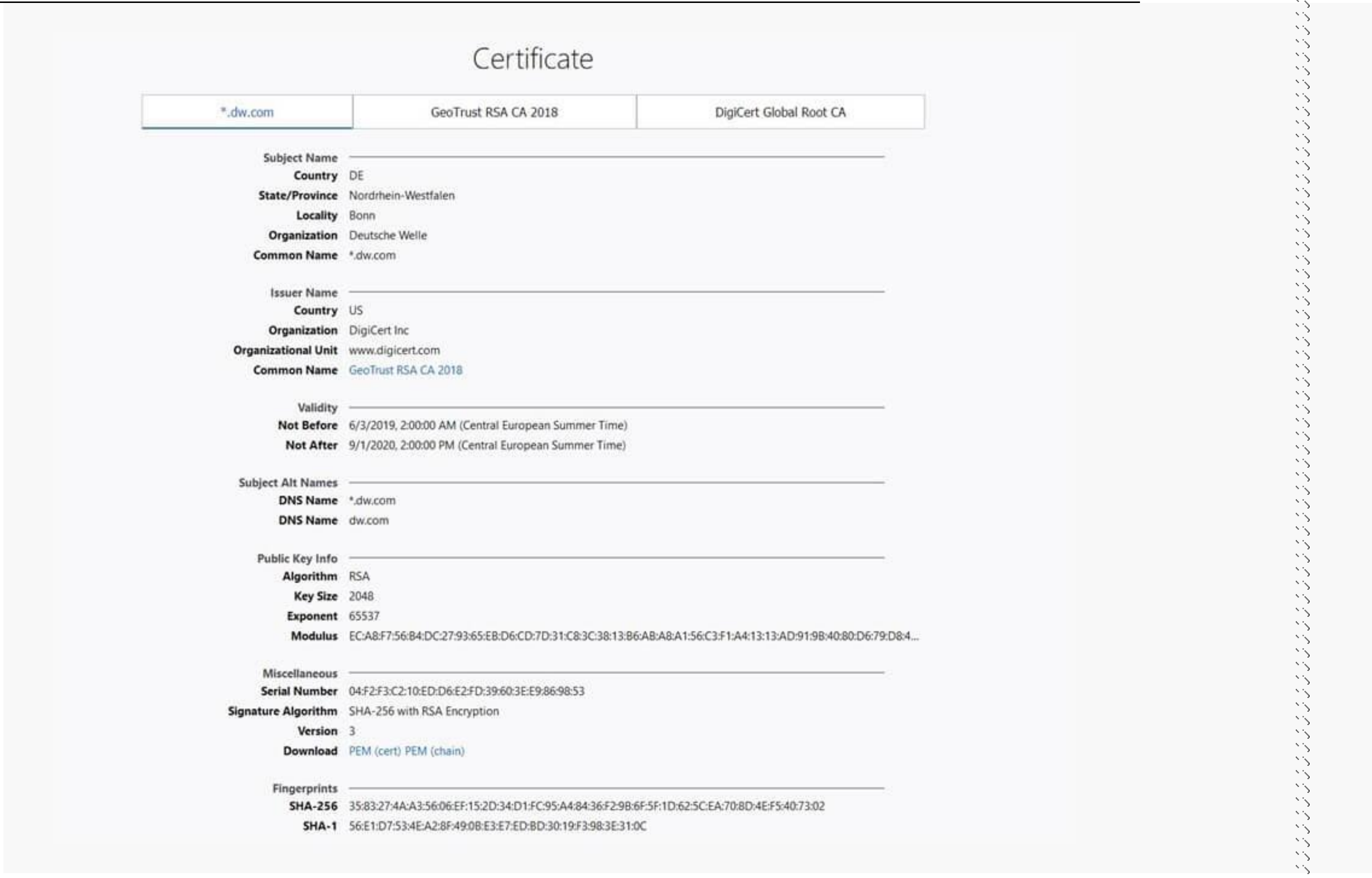
```
<?php
```

```
echo hash('sha256', 'apple');
?>
```

Here, the “sha256” encryption algorithm is being used to hash the input value “apple”. The corresponding hash value or fingerprint is always “3a42c503953909637f78dd8c99b3b85ddde362415585afc11901bdefe8349102”.

Hash functions and websites

With SSL-encrypted data transmission, when the web server receives a request, it sends the server certificate to the user’s browser. A session ID is then generated using a hash function, and this is sent to the server where it is decrypted and verified. If the server approves the session ID, the encrypted HTTPS connection is established and data can be exchanged. All of the data packets exchanged are also encrypted, so it is **almost impossiblefor hackers to gain access**.



Certificate

Subject Name	
Country	DE
State/Province	Nordrhein-Westfalen
Locality	Bonn
Organization	Deutsche Welle
Common Name	*.dw.com
Issuer Name	
Country	US
Organization	DigiCert Inc
Organizational Unit	www.digicert.com
Common Name	GeoTrust RSA CA 2018
Validity	
Not Before	6/3/2019, 2:00:00 AM (Central European Summer Time)
Not After	9/1/2020, 2:00:00 PM (Central European Summer Time)
Subject Alt Names	
DNS Name	*.dw.com
DNS Name	dw.com
Public Key Info	
Algorithm	RSA
Key Size	2048
Exponent	65537
Modulus	EC:A8:F7:56:B4:DC:27:93:65:EB:D6:CD:7D:31:C8:3C:38:13:B6:AB:A8:A1:56:C3:F1:A4:13:13:AD:91:9B:40:80:D6:79:D8:4...
Miscellaneous	
Serial Number	04:F2:F3:C2:10:ED:D6:E2:FD:39:60:3E:E9:86:98:53
Signature Algorithm	SHA-256 with RSA Encryption
Version	3
Download	PEM (cert) PEM (chain)
Fingerprints	
SHA-256	35:83:27:4A:A3:56:06:EF:15:2D:34:D1:FC:95:A4:84:36:F2:9B:6F:5F:1D:62:5C:EA:70:8D:4E:F5:40:73:02
SHA-1	56:E1:D7:53:4E:A2:8F:49:0B:E3:E7:ED:BD:30:19:F3:98:3E:31:0C

An extract from the certificate for German broadcasting corporation Deutsche Welle, showing the key the server uses to establish a communication session with the user’s browser.

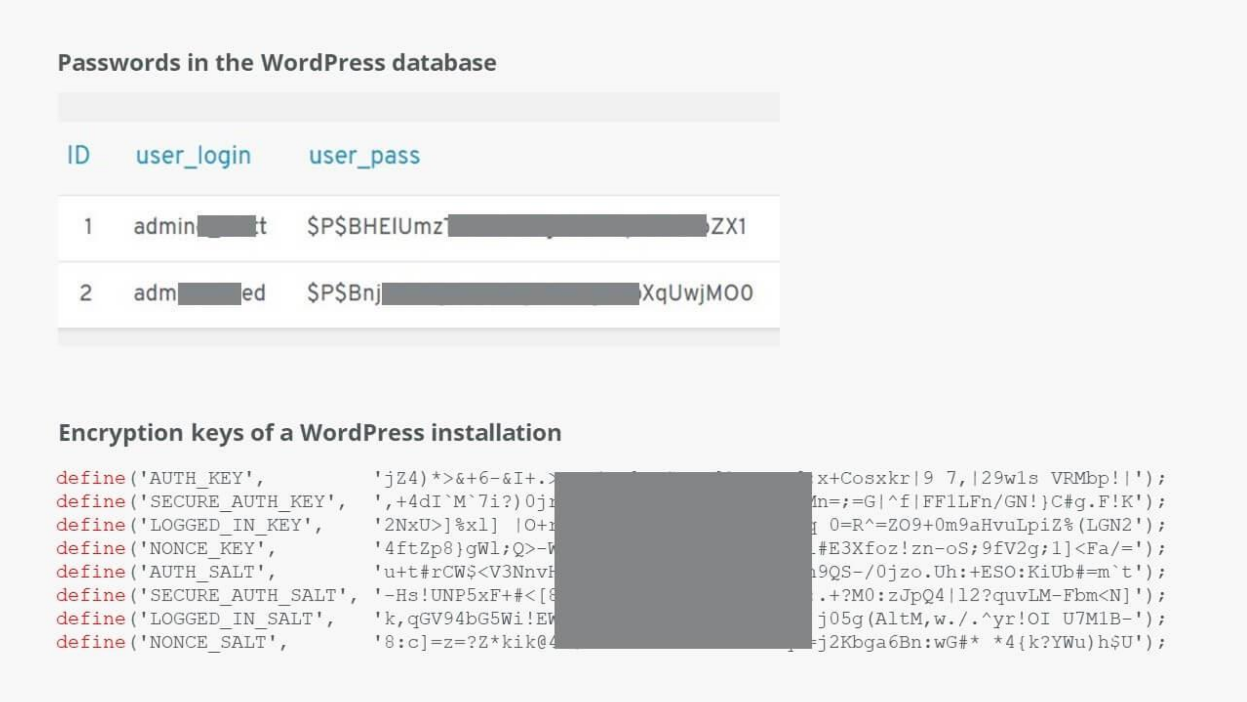
Session IDs are generated using data relating to a site visit, such as the IP address and time stamp, and communicated with the URL. One common use of session IDs is to give unique identifiers to people shopping on a website. Nowadays, session IDs are rarely passed as a URL parameter (for example, as something like *www.domain.tld/index?sid=d4ccaf2627557c756a0762419a4b6695*). Instead, they are stored as a **cookie** in the website header.

Hash values are also used to encrypt **cached data** to prevent unauthorized users from using the cache to access login and payment details or other information about a site.

Communication between an **FTP server and a client** using the SFTP protocol also works in a similar way.

Protection of sensitive data

Login details for online accounts are frequently the target of **cyber-attacks**. Hackers either want to disrupt operation of a website (for example, to reduce income generated by traffic-based ads) or access information about payment methods.



Passwords in the WordPress database

ID	user_login	user_pass
1	admin[redacted]t	PBHEIUmz[redacted]ZX1
2	adm[redacted]ed	PBnj[redacted]XqUwjMO0

Encryption keys of a WordPress installation

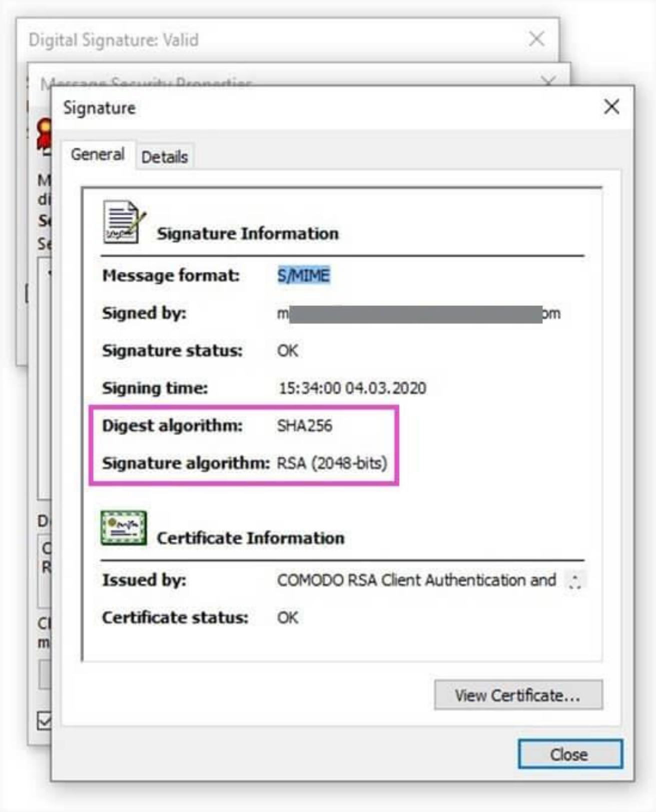
```
define('AUTH_KEY',         'jZ4)*>&+6-&I+.[redacted]x+Cosxkr|9 7,|29w1s VRMbp!|');
define('SECURE_AUTH_KEY',  ',+4dI`M`7i?)0j           Mn=;=G|^f|FF1LFn/GN!}C#g.F!K');
define('LOGGED_IN_KEY',    '2NxU>]%xl] |O+            0=R^=ZO9+0m9aHvuLpiZ%(LGN2');
define('NONCE_KEY',        '4ftZp8}gWl;Q>-            #E3Xfoz!zn-oS;9fV2g;1]<Fa/=');
define('AUTH_SALT',        'u+t#rCW$<V3Nnv            9QS-/0jzo.Uh:+ESO:KiUb#=m`t');
define('SECURE_AUTH_SALT', '-Hs!UNP5xF+#<[            .+?M0:zJpQ4|12?quvLM-Fbm<N]');
define('LOGGED_IN_SALT',   'k,qGV94bG5Wi!E            j05g(AltM,w./.^yr!OI U7M1B-');
define('NONCE_SALT',       '8:c]=z=?Z*kik@            j2Kbga6Bn:wG#* *4{k?YWu)h$U');
```

The WordPress Content Management System offers a range of security functions for authenticating registered site users. The keys shown above were generated using various hashing algorithms.

In the WordPress example above, you can see that **passwords are always encrypted** before they are stored. Combined with the session IDs generated in the system, this ensures a high level of security. This is especially important for **protection against "brute force attacks"**. In this kind of attack, hackers use their own hash functions to repeatedly try out combinations until they get a result that allows them access. Using long passwords with high security standards makes these attacks less likely to succeed, because the amount of computing power required is so high. Remember: **Never use simple passwords, and be sure to protect all of your login details and data against unauthorized access.**

Digital signatures

Email traffic is sent via servers that are specially designed to transmit this type of message. Keys generated using hash functions are also used to add a digital signature to messages.



Message format: S/MIME

Signed by: [redacted]

Signature status: OK

Signing time: 15:34:00 04.03.2020

Digest algorithm: SHA256

Signature algorithm: RSA (2048-bits)

Issued by: COMODO RSA Client Authentication and

Certificate status: OK

Adding a digital signature to an email is like signing a handwritten letter – you sign once, and your signature is unique.

The steps involved in **sending an email** with a **digital signature** are:

- Alice (the sender) converts her message into a hash value and encrypts the hash value using her private key. This encrypted hash value is the digital signature.
- Alice sends the email and the digital signature to the recipient, Bob.
- Bob generates a hash value of the message using the **same hash function**. He also decrypts the hash value using Alice's public key and compares the two hashes.
- If **the two hash values match**, Bob knows that Alice's message has **not been tampered** with during transmission.

Please note that a digital signature proves the integrity of a message but does not actually encrypt it. If you're sending confidential data, it's therefore best to encrypt it as well as using a digital signature

How can hash functions be used to perform lookups?

Searching through **large quantities of data** is a very resource-intensive process. Imagine you've got a table listing every inhabitant of a big city, with lots of different fields for each entry (first name, second name, address, etc.). Finding just one term would be very time-consuming and require a lot of computing power. To simplify the process, each entry in the table can be converted into a unique hash value. The search term is then converted to a hash value. This limits the number of letters, digits and symbols that have to be compared, which is much more efficient than searching every field that exists in the data table, for example, for all first names beginning with "Ann".

Summary

Hash functions are used to improve security in electronic communications, and lots of highly sophisticated standards have now been developed. However, hackers are aware of this and are constantly coming up with more advanced hacking techniques.