Hash Functions

A *hash function* is an algorithm which converts a message into a simple data value by chopping up and scrambling the bits or bytes that make up the message. The output from the function, called the hash value, is normally of a reasonably small fixed size.

Hash functions have lots of uses, and there are lots of different approaches to creating a hash function depending exactly what you want to use it for.

Some of the properties which are generally good for hash functions – especially if the hash function is being used for security:

- **One-way:** Once a hash value has been created from a message, it must be impossible to convert it back to the original message.

  For example instead of storing your password in a database, a web site should always store the hash value of your password. The stored hash value can be used to check your password when you long in; the web server will calculate the hash value of the password you typed and compare it to the hash value stored in the database. If the passwords were the same, then the hash values will also be the same.

  The "one-way" property means that any hacker who manages to see the web site's database won't be able to work out what your password was and use it on a different web site where you have reused passwords. (Even so, don't reuse passwords!)

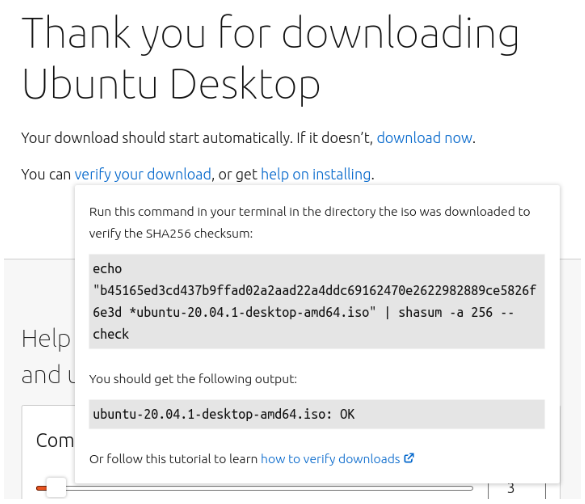
- **Collision-Free:** It should be infrequent, or exceedingly rare, that two different messages produce the same hash value. When two messages result in the same hash value, this is called a "Collision".

  In the example where hashed passwords are stored in the database, collisions would allow two or more different passwords to match the hash value in the database. When using hashes for security, the algorithm should make it virtually impossible that two different inputs hash to the same output.

Another use of hash functions is to calculate a checksum for a message. A checksum is a value which can be easily compared to ensure that a message has not been accidentally corrupted or intentionally tampered with.

For example, when you download Ubuntu Linux the web site shows the SHA256 (Secure Hash Algorithm 256 bit) checksum, so you can check that the file you have downloaded is genuine and has not been modified with viruses, etc.

If a hacker was able to insert a virus and at the same time ensure that the SHA256 checksum had not changed, that would be bad!



Thank you for downloading Ubuntu Desktop

Your download should start automatically. If it doesn't, download now.

You can verify your download, or get help on installing.

Run this command in your terminal in the directory the iso was downloaded to verify the SHA256 checksum:

```
echo
"b45165ed3cd437b9ffad02a2aad22a4ddc69162470e2622982889ce5826f
6e3d *ubuntu-20.04.1-desktop-amd64.iso" | shasum -a 256 --
check
```

You should get the following output:

```
ubuntu-20.04.1-desktop-amd64.iso: OK
```

Or follow this tutorial to learn how to verify downloads

Rebel Alliance Checksum (RA-2)

The Rebel Alliance has created a hash function algorithm which can be used to calculate a 2 digit checksum for a message made up of letters of the alphabet. Letter ‘A’ has value 1 and ‘Z’ is 26.

The algorithm works like this:

```
Set checksum = 0
Repeat for each letter in the message:
    Set checksum = (checksum * checksum) + (letter value)
                                  [take the last 2 digits only]
```

Here is an example of how to use this algorithm to calculate the checksum of the word “HELLO”.

1. To start off we always set checksum = 0
2. Process each letter one at a time:
   a) Set checksum = 0 * 0 + 8 (**H**) = 8
   b) Set checksum = 8 * 8 + 5 (**E**) = 69
   c) Set checksum = 69 * 69 + 12 (**L**) = 4773 → last two digits only = 73
   d) Set checksum = 73 * 73 + 12 (**L**) = 5341 → last two digits only = 41
   e) Set checksum = 41 * 41 + 15 (**O**) = 1696 → last two digits only = **96**

We used the algorithm to calculate that the checksum value of the word “HELLO” is **96**.

Alice, your friend in the Rebel Alliance, has sent you a some words, each of which is accompanied by a valid checksum. The words make up a message. At the same time the Empire is sending out random words and numbers as radio interference to confuse you; these numbers are not valid checksums.

These are all the words and numbers you received but you don’t know who sent each word.

HELLO	96
ALICE	74
PZERO	94
EMPIRE	96
REBEL	88
WINS	81
LOSES	6
LOVES	15
BATTLE	74
YOU	12
BROWN	37

Which are the words from Alice which have the correct checksum? What is the message?

The Rebel Alliance hash function should definitely not be used for security! Bonus questions:

1. Secure hash functions should avoid collisions, but our hash function isn’t very good at that. **Which of the two words above have the same hash value?**

2. Secure hash functions should make it hard for a hacker to create a new message with the same hash value as something else. **Can you find a word with a hash value of 37?**