

Re: password length

Source: <http://linux.derkeiler.com/Newsgroups/alt.os.linux.suse/2004-06/4948.html>

From: Bill Unruh (unruh_at_string.physics.ubc.ca)

Date: 06/27/04

Date: Sun, 27 Jun 2004 20:13:46 +0000 (UTC)

Michael W. Cocke <cocke@catherders.com> writes:

]On Sun, 27 Jun 2004 16:37:11 +0000 (UTC), unruh@string.physics.ubc.ca

](Bill Unruh) wrote:

]>Michael W. Cocke <cocke@catherders.com> writes:

]>

]>]On Sun, 27 Jun 2004 12:24:00 +0200, Hans Schilling

]>]<news@compuserve.de> wrote:

]>

]>]>But what is the different between DES and MD5 ?

]>]>

]>]>Thanks.

]>]>

]>]>Hans.

]>

]>]The short answer (for non-cryptographers) is "Different encryption

]>]methods".

]>

]>]It is NOT encryption. It is hashing -- reducing an input into a random

]>]seeming output of fixed length-- 64 bits in teh case of the crypt(3) DES

]>]based hash, 128 bits in the case of the MD5 based hash.

]>

]>]More detail follows for those who care -

]>

]>]DES is the US Data Encryption Standard, which was broken a few years

]>]ago but takes much horsepower to reverse.

]>

]>]It has not been broken. Rather its password length is too short, and trying

]>]every password until one works can now be done.

]>

]>]DEs is not used as a crypto -- instead it is used as a hash of the key.

]>]-- zero is encrypted 25 times with the password, and the output is the hash

]>]of the password. It cannot be reversed.

]>

]>

]>

]>

]>]Message Digest ver 5 (MD5) takes as input a message of arbitrary
]>]length and produces as output a 128-bit "fingerprint" or "message
]>]digest" of the input.
]>
]>]MD5 is NOT used per se in the MD5 based hash. Instead MD5 is reused
]>]multiple times with further bit rotations, shiftings,... in order to slow
]>]down the hashing.
]>
]>]The MD5 algorithm is actually intended for digital signature
]>]applications, not really crypto at all, but it works acceptably
]>]because it's computationally infeasible to produce two messages having
]>]the same message digest. In principal, similar to a CRC32.
]>
]>]MD5 is a hash, and it is used as a hash in the password algorithm.
]>]It is NOT used as an encryption. There is no way, no password and no
]>]algorithm, which will take the entry in the /etc/passwd or /etc/shadow file
]>]and create the password again.
]>]Ie, the password algorithms are not encryptions, they are hashes. And they
]>]work by hashing the entered password and seeing if it matches the original.
]>]Thus there is some probability that more than one password would equally
]>]well on any account. Finding even one is however very very difficult.
]>
]>]The advantage of the MD5 based passwords is that instead of 8 characters,
]>]they effectively use 16 for the password (it can use an arbitrary length
]>]password stream, but the probability is very high that there exists some
]>]16 character(8 bit character) password which would also work.) This makes
]>]exhaustive search much more difficult.
]>
]>
]>
]>
]>]Mike--
]>
]>]--
]>]If you're not confused, you're not trying hard enough.
]>]--
]>
]>]If you are not confusing others, you are not trying hard enough.

]Bill, calling something a hash and acting as though that's a magic
]word doesn't make you correct. My knowledge of the subject of data
]reduction and encryption is a bit dated, but at one point I knew a
]fair bit about it, and I stand by what I said.

No. But accurately describing what happens does make me correct.

a hash for those not in the know is a way of representing an arbitrary data
stream as a fixed length piece of data. A cryptographic hash is one where it
is very difficult to find some input stream which will give the given
output. There is no way to reverse a hash-- to find the input from the
output-- except exhaustive search.

Encryption is a one to one function which takes an input and makes an output in such a way that it is very difficult from the output to figure out the input, unless one has the key, in which case it is easy. Hashes are not one to one. Encryptions are easy to reverse if you know the key, hashes cannot be reversed.

All of the password techniques, crypt(3) and MD5 based are cryptographic hashes.

The password is hashed and the hash output stored in the passwd or shadow file. When a user tries to log on, his password is hashed and that hash is compared with the stored one. If they are the same, then the password is assumed correct.

There is no way of figuring out the password from the stored hash, not even for root, other than by exhaustive search. There is no key to decrypt the stored passwords. It is not an encryption, it is a hash.

Note that as an added feature, all password hashes also use what is called a salt. This is an additional, random, known piece of text which alters the hash, so that an attacker cannot easily make up a table of hashes of many possible passwords. The storage in the password record in /etc/passwd or shadow has the first n (3 for crypt(3) old style Unix hash and I think 5 for the newer MD5 based hashes) bytes as that salt, and the rest the password as hashed with that salt.