MD5Online

Best source of MD5 information

3 Reasons why MD5 is not Secure

You probably already read that information, and you know that MD5 is not the most secure hashing function
But do you know why? Do you know safer alternatives?
This is what I'll explain you today

Why MD5 is not secure?
MD5 is a cryptographic algorithm, often used to store passwords in a database
But this algorithm is no longer safe
Brute force attacks are faster than ever, dictionary tables are big and there are other potential problems with the MD5 algorithm

I'll explain all of this in this article

Table of Contents

What is MD5?

MD5 is a cryptographic algorithm, often used to store passwords in a database
At the beginning of the Internet, websites mainly kept passwords in clear text in their databases
That was not a good option, so people used MD5 to obfuscate the password in the database

MD5 is an algorithm that produce a 32 characters hexadecimal string from any password, phrase or text
For example, if your password is 'qwerty' (bad idea), in the database you'll have d8578edf8458ce06fbc5bb76a58c5ca4

That way, IT staff can't see your password, and if someone stole the database, they don't get all the passwords directly
Today, it's still not immediate to decrypt passwords, but not so far
I'll explain why in the next parts, and why you must find another way to store passwords

Why MD5 is not secure (3 reasons)

1 – Brute force attacks are fast

A brute force attack is a way to find a password by trying a lot of possibilities
Either by guessing what the user could have used (birth date, the child's names, pet names, …), or by trying everything (from a,b,c to 10 characters passwords with special characters)

The MD5 algorithm is fast to use
So in a few seconds you can try many combinations

20 years ago it could take years to find a password for the world's most powerful computers
Today, everyone has a super-computer at home, with improvements in the processor and graphics processor, we can decrypt "secure" passwords in a few days maximum.
The best computers can try billions of passwords every second (source: ZDNet)

The only resistance to the brute force attacks are probably the password length
If you have a 40 characters long random password, with special characters, you're probably safe for the moment
But for how much longer?

2 – Dictionary tables are big

On MD5Online we like dictionary tables
By storing over 1,150 billions passwords in our database, we can give you an answer in a few seconds for any hash

That's the second problem with the MD5 algorithm
It is so widely used that huge databases like this have been created over the years
If your password is inside (and there is a good chance if you have a "short" password), your accounts are not safe at all

As for the brute force method, the only way to be safe is to use a long random password with special characters
There are too much possibilities to have it in this kind of database
Database like this are taking a lot of disk space. Even if it's cheaper and cheaper over the years, it's still an obstacle

3 – Collisions

The MD5 algorithm has also proven issues within its cryptographic method
A collision is when two words have the same hash generated

Safe algorithms have a good collision resistance
That's to say that you have low chances to get the same hash for different words
But MD5 has a low collision resistance

So if you know that "abc" and "def" have the same generated hash (just an example)
You can say that "123abc" and "123def" have also the same hash generated
And this is a bad property for a cryptographic hash functions as you can guess a lot of derived words

What are the solutions?

Now that you know why MD5 is not safe, what can you do to improve your database security

Use salt

The first thing you can try is to use salt while encrypting passwords
I already wrote an article about this: What is an MD5 salt and how to use it?
Check it if you want to learn more about this

Basically, a salt is a word you'll add before and/or after each password
If your salt is "randomsaltformypassword" and the user choose "qwerty" as a password
You'll use "randomsaltformypasswordqwerty" as the MD5 function parameter

That way you are encrypting a much longer password in your database, and it will be harder for a hacker to find the corresponding password
Make sure to choose a long salt to improve security enough

Long passwords

Another solution is to force users to use longer password (maybe 15 characters or more)
You can also add passwords complexity to make sure they are using uppercase, lowercase and special characters

But be careful, people will often use weak passwords, even if you implement all of this
"ILoveMyCompany!" is a 15 character password with a special character, but it's easy to guess

Or even worse, they will note the password on a post-it note near the computer:)

Other hash functions

Probably the best solution is to use another cryptographic algorithm
This is not the easiest because you probably have to change your database structure, but it could be the safest

I'll not give you too many examples as if you are reading these lines in ten years it could have changed
But today, the password_hash() function in PHP seems a good idea (check the documentation).
Or maybe bcrypt or scrypt with a salt and enough iterations are also a good solution

Conclusion

That's it, you now know why the MD5 algorithm is no longer safe to use for password encryption
And I also give you other alternatives to improve your database security

Try to keep up to date with the latest security news, there are new breaches every day, and this article can quickly become obsolete