

Forum: [Hack Forums](#)

Thread: [CRT Hashing Algorithm](#)

zeroexp - June 12, 2014, 1:03 pm

Hello community, i came out with something new today.. a hashing algorithm. Of course it may need a little bit more work so don't be hateful if you reply... After all , nothing's perfect. Let's get to it.

It took a lot of time for me to think how to make it irreversible and finally figured it out. In my opinion I think I did a pretty good job as the hash depends on a lot of different factors. ?

I made a program for the hashing process AND a DLL so you can use it yourself . Let's start.

[CLICK ME FOR THE ACTUAL VT SCAN](#)

[CLICK ME FOR THE EXECUTABLE FILE](#)

[CLICK ME FOR THE DLL](#)

How does the app look

[Video: <https://www.youtube.com/watch?v=FrM5c77kq68>]

How to add the DLL to your app

[CLICK ME](#)

Thanks for reading ! ? Leave a reply if you think something is wrong , have a problem or have a suggestion. I read everything because I'm still trying to learn... I'm a newbie in VB.NET :P

+Haze - June 12, 2014, 1:49 pm

Looks nice, I would like to use this but, I don't see anyway in which it would be useful? Apart from hashing passwords.

ubbelol - June 12, 2014, 1:49 pm

Do you really need such a big thread and everything for just this? :p Either way, always good to see releases. But why are you returning an "object" when it's always a string? Also is there some documentation about this hash somewhere? I haven't heard of it before.

zeroexp - June 12, 2014, 1:52 pm

@Haze - I added you

@Ubbelol - Sorry about that object thingie :P You never heard of it because i just made it and posted it here...

Tried to do sth new so i can learn more. And, maybe someone finds this useful.

Zettabit - June 12, 2014, 2:34 pm

Is there any reason you used a Obfuscator on both of these downloads?

EDIT: This is also pretty insecure as I can already see a pattern.

EDIT2: I'll rephrase that a little, it's insecure with < 5 letters

tweety - June 12, 2014, 2:36 pm

Is this hash algorithm created by you or is it an existing algorithm?

zeroexp - June 12, 2014, 9:42 pm

@Tweety. - I created it ... I just chose a random 3 letter name because that's how most algorithms are called

@Zettabit - It's not really an obfuscator but i got used to do that every time even if i don't post a program. Well, i don't know , i will try to update this if i can and make it better but i have my final exams in a week.

Loladmin - June 12, 2014, 10:26 pm

I don't know why you did this:

```
Code string[] strArray = new string[] {  
    "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p",  
    "r", "s", "t", "u", "v", "x", "y", "z", "a", "b", "c", "d", "e", "f", "g", "h",  
    "i", "j", "k", "l", "m", "n", "o", "p", "r", "s", "t", "u", "v", "x", "y", "z",  
    "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p",  
    "r", "s", "t", "u", "v", "x", "y", "z", "a", "b", "c", "d", "e", "f", "g", "h",  
    "i", "j", "k", "l", "m", "n", "o", "p", "r", "s", "t", "u", "v", "x", "y", "z",  
    "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p",  
    "r", "s", "t", "u", "v", "x", "y", "z", "a", "b", "c", "d", "e", "f", "g", "h",  
    "i", "j", "k", "l", "m", "n", "o", "p", "r", "s", "t", "u", "v", "x", "y", "z",  
    "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p",  
    "r", "s", "t", "u", "v", "x", "y", "z", "a", "b", "c", "d", "e", "f", "g", "h",  
    "i", "j", "k", "l", "m", "n", "o", "p", "r", "s", "t", "u", "v", "x", "y", "z",  
    "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p",  
    "r", "s", "t", "u", "v", "x", "y", "z"  
};
```

But it just seems unnecessary. You could've just created a string, because technically strings are character arrays.

Anyways, no offence, but it seems poorly coded. It looks like you spent more time on the GUI than

the actual function itself.

zeroexp - June 13, 2014, 1:39 am

As i said i am still a beginner and i did not know how to do that... I tried my best...

mcly - June 13, 2014, 5:01 am

Would be cool if you added some information to this as of how this works, why you choose to code on this

What advantages it has over other hashing algorithms, etc..

I honestly think though that it has a way bigger collision ratio, is slower, and might be reversible..

Im not trying to be mean, but why did you create this if all other hashing methods are just better?

I see no point in someone using this, if other methods are just better...

zeroexp - June 13, 2014, 5:16 am

Did it so i can learn more about this stuff. You learn a lot from practice so ... i just tried to do sth different from what i did before. ^_^ I also wanted some replies because i know HF has some of the best critics ever xD You're not being mean, i enjoy reading every reply.

Thank you.

TheBarret - June 13, 2014, 12:08 pm

Ehh....sorry mate this looks not that good.

?XilluX - June 13, 2014, 12:55 pm

Well a hash has for example always 16, 32, or 64 char. Your hash does always have another length.

SIM-1456245 - June 13, 2014, 1:03 pm

(Jun 13, 2014 12:55 PM)?XilluX Wrote: [\[CITE\]](#) Well a hash has for example always 16, 32, or 64 char. Your hash does always have another length.

That's called a feature of the algorithm.

BTW The name CRT reminds me on an old CRT TV.

LordCoder - June 13, 2014, 3:02 pm

Your hashing function can be partially reversed. It isn't a perfect hashing function as it just converts and deletes some parts of the original string. But you can test it with the same kind of letter.

Hash of "a": bz791

Hash of "aa": bz791b91 (just added b91 lol)

This can cause a collision between 2 strings. And this hashing doesn't have the avalanche effect which is normally seen in SHA or MD5 as you see above.

Using Math.Round you just can make it "irreversible" but using a dictionary is most like an encoding.

EDIT: Found a collision with your hash.

Hash of "abcd": n2z71d41c7b91

Hash of "azcd": n2z71d41c7b91

Used a basic brute forcer I've just created: <https://www.firedrive.com/file/B346C237C6BB9FEC>

[img]http://s25.postimg.org/hk3qxhvp3/2014_06_14_00_14_43_2014_06_14_00_14_07_Windows.png[/img]

shawnw1104 - August 15, 2014, 9:33 am

That is the most beautiful GUI i have ever seen :)

Just my two-cents is that this is a good starting point, but needs major improvements (if not rewritten).

You should be processing things in a fixed ammount. Lets say in 64 bytes at a time. From there create 4

16 byte arrays from that 64 chunk and do some mixing (aka. diffusion). Place that into a for() and do 4 rotations. Every time make sure its not hashing the same thing.

example:

if we have A,B,C,D. We will loop 4 times so every part depends on the rest of the blocks. A = B, B = C, C = D, D = A. or something like that. The point being is that you wanna be dependent on EVERY byte. using << and >> are handy but used incorrectly will overflow things. do, String -> Byte[] -> convert to Int[] to use << or >>.

XoR comes in handy. Not sure how to do it in vb.net. C# its ^. Read up on XoR first because its good to know how it works.

The output of the hash function will be 512 bits (512 / 8)=64 bytes long.

Having a Avalanche Effect that is effective is hard to do. Your best bet is to look at a function like MD5. Use Bouncy Castle Library for source if need be.

Create a function that creates a key for the current block (i would use RC5 since its nice and patent free. just modify it a bit).

Best thing you can honestly do is study current hashing algorithm (skein/tiger/RIPEMD/Whirlpool/etc). Its the best way to see how they do it.

Just a piece of advice. Buffer.BlockCopy() is your best friend ;)

...

Shawn W.

zeroexp - August 15, 2014, 10:33 am

(Aug 15, 2014 09:33 AM)shawnw1104 Wrote: [\[CITE\]](#) That is the most beautiful GUI i have ever seen :)

Just my two-cents is that this is a good starting point, but needs major improvements (if not rewritten).

You should be processing things in a fixed ammount. Lets say in 64 bytes at a time. From there create 4

16 byte arrays from that 64 chunk and do some mixing (aka. diffusion). Place that into a for() and do 4 rotations. Every time make sure its not hashing the same thing.

example:

if we have A,B,C,D. We will loop 4 times so every part depends on the rest of the blocks. A = B, B = C, C = D, D = A. or something like that. The point being is that you wanna be dependent on EVERY byte. using << and >> are handy but used incorrectly will overflow things. do, String -> Byte[] -> convert to Int[] to use << or >>.

XoR comes in handy. Not sure how to do it in vb.net. C# its ^ . Read up on XoR first because its good to know how it works.

The output of the hash function will be 512 bits (512 / 8)=64 bytes long.

Having a Avalanche Effect that is effective is hard to do. Your best bet is to look at a function like MD5. Use Bouncy Castle Library for source if need be.

Create a function that creates a key for the current block (i would use RC5 since its nice and patent free. just modify it a bit).

Best thing you can honestly do is study current hashing algorithm (skein/tiger/RIPEMD/Whirlpool/etc). Its the best way to see how they do it.

Just a piece of advice. Buffer.BlockCopy() is your best friend ;)

...

Shawn W.

Lol this thread is pretty old. Now i got better at VB.NET , i'll give it a try later. Imma actually try in the future to make one again but way better.

Thanks for the reply. People helped me a lot understanding a hashing algorithm.

dR1234 - August 19, 2014, 1:16 pm

What is the theme name pleaseeee? Thanks you

Odus - August 20, 2014, 6:44 am

(Aug 19, 2014 01:16 PM)dR1234 Wrote: [\[CITE\]](#) What is the theme name pleaseeee? Thanks you

Pay attention boy!