

Forum:[Hack Forums](#)

Thread: [StringEncrypt - String & File Encryption for VB.NET](#)

PELock - February 20, 2019, 3:59 pm

String and file encryption for VB.NET developers. StringEncrypt page allows you to encrypt strings and files using randomly generated algorithm, generating a unique decryption code (polymorphic code) in the selected programming language. Simple & fast - try it yourself!

Example:

Code ' encrypted with StringEncrypt (v1.3.0) [Visual Basic .NET]

' EncryptedString = "Visual Basic .NET string encryption"

Dim epTlz() As Integer = { &H8ED9, &H3BD9, &HF1D8, &HAFD8, &H63D8, &H38D8, &H44D8, &HC2D7,

&H85D7, &H33D7, &HFDD6, &HA3D6, &HC4D6, &H76D6, &H16D6, &HC1D5,
&H90D5, &H84D5, &HF1D4, &HB0D4, &H76D4, &H3FD4, &HFAD3, &HBDD3,
&HC4D3, &H21D3, &HFAD2, &HC7D2, &H76D2, &H2DD2, &HF4D1, &HB0D1,
&H63D1, &H35D1, &HF6D0 }

Dim EncryptedString As String

Dim BePuQ As Integer

Dim blvjn As Integer

For blvjn = 0 To 34

BePuQ = epTlz(blvjn)

BePuQ -= 1

BePuQ = (((BePuQ And &HFFFF) >> 13) Or (BePuQ << 3)) And &HFFFF

BePuQ += blvjn

BePuQ -= &HF74C

BePuQ += blvjn

BePuQ -= &H2E7B

BePuQ = (((BePuQ And &HFFFF) >> 8) Or (BePuQ << 8)) And &HFFFF

BePuQ = Not BePuQ

BePuQ = BePuQ Xor blvjn

BePuQ += 1

BePuQ += blvjn

BePuQ = ((BePuQ << 13) Or (BePuQ And &HFFFF) >> 3) And &HFFFF

EncryptedString = EncryptedString + ChrW(BePuQ And &HFFFF)

Next blvjn

MessageBox.Show(EncryptedString)

WebApi

[h**ps://github.com/PELock/StringEncrypt-WebAPI](https://github.com/PELock/StringEncrypt-WebAPI)

0xDEAD10CC - February 21, 2019, 4:06 pm

I don't really see any great benefit here truthfully. By the very looks of that VB.NET code too even though I have barely touched VB.NET in 5 or more years, it's sloppily generated. If you rely on this and obfuscation to protect your strings I'd happily dump every last secret string in your binary with ease..

PELock - February 22, 2019, 5:30 am

1. Benefits - to automate string encryption (via WebAPI)
2. Code is beautifully generated (show me better if you say it's 'sloppily generate')
3. How else would you like to encrypt generic strings? This one is polymorphic.

0xDEAD10CC - February 22, 2019, 10:00 am

(Feb 22, 2019 06:30 AM)PELock Wrote: [\[CITE\]](#) 1. Benefits - to automate string encryption (via WebAPI)

2. Code is beautifully generated (show me better if you say it's 'sloppily generate')
3. How else would you like to encrypt generic strings? This one is polymorphic.
1. It's a pretty weak encryption (of which won't really prevent me from cracking your application)
2. I'm not talking about code formatting
3. There's lots of better methods I've seen already built into obfuscators

Any other noob questions?

PELock - February 22, 2019, 6:44 pm

1. It's a polymorphic string encryption, not AES, RC4 or ECC, it's not meant to be anti-cracking solution (who said that?)

2. So what are you talking about? It's fast, supports both strings and files, all arithmetic and bitwise operations are supported (xor, not, neg, rol, ror, shl, shr), supports both ansi/unicode, checks for every language keyword to not generate random label with the name of the restricted keyword, has Windows, online and PHP clients.

3. StringEncrypt supports C/C++, Delphi/Pascal, Java, JS, Ruby, Python, Haskell, MASM, FASM, VB.NET, C#, PowerShell, do you have obfuscators for all of them?

BTC - February 22, 2019, 6:53 pm

(Feb 22, 2019 11:00 AM)0xDEAD10CC Wrote: [\[CITE\]](#) 1. It's a pretty weak encryption (of which won't really prevent me from cracking your application)

2. I'm not talking about code formatting
3. There's lots of better methods I've seen already built into obfuscators

Any other noob questions?

I'm not sure you actually know what the OP posted here. The code is good, I have ideas to add but am too tired do real testing. I just don't see why you comment complaining about something then show in every single way possible you have no idea what this is. Its for polymorphic string encryption why are you here talking about cracking his "application". You troll

Ajumma - February 22, 2019, 7:00 pm

(Feb 22, 2019 07:44 PM)PELock Wrote: [\[CITE\]](#) 1. It's a polymorphic string encryption, not AES, RC4 or ECC, it's not meant to be anti-cracking solution (who said that?)

2. So what are you talking about? It's fast, supports both strings and files, all arithmetic and bitwise operations are supported (xor, not, neg, rol, ror, shl, shr), supports both ansi/unicode, checks for every language keyword to not generate random label with the name of the restricted keyword, has Windows, online and PHP clients.

3. StringEncrypt supports C/C++, Delphi/Pascal, Java, JS, Ruby, Python, Haskell, MASM, FASM, VB.NET, C#, PowerShell, do you have obfuscators for all of them?

Just curious, if it's not meant to be an anti-cracking method, why use it? What's the benefit of using something like this?

It prevents a normal noobie from looking at my program and seeing the plaintext password, which

you should not do in the first place anyways, but if it's not irreversible then what's the point of using it?

I'm just a noob, so please explain :D

0xDEAD10CC - February 22, 2019, 10:21 pm

(Feb 22, 2019 07:53 PM)Apex Wrote: [\[CITE\]](#) I'm not sure you actually know what the OP posted here. The code is good, I have ideas to add but am too tired do real testing. I just don't see why you comment complaining about something then show in every single way possible you have no idea what this is. Its for polymorphic string encryption why are you here talking about cracking his "application". You troll

I know exactly what they posted which is the only reason why I replied to the thread. This is basic is what it is, and if it wasn't bad code then the encrypted string variable would be initialized to an empty string before concatenation.

```
Dim EncryptedString As String = String.Empty
```

(Feb 22, 2019 07:44 PM)PELock Wrote: [\[CITE\]](#) 1. It's a polymorphic string encryption, not AES, RC4 or ECC, it's not meant to be anti-cracking solution (who said that?)

2. So what are you talking about? It's fast, supports both strings and files, all arithmetic and bitwise operations are supported (xor, not, neg, rol, ror, shl, shr), supports both ansi/unicode, checks for every language keyword to not generate random label with the name of the restricted keyword, has Windows, online and PHP clients.

3. StringEncrypt supports C/C++, Delphi/Pascal, Java, JS, Ruby, Python, Haskell, MASM, FASM, VB.NET, C#, PowerShell, do you have obfuscators for all of them?

Nobody uses all of those languages all at once. I only need one good one for the language I'm using.

All of this would be relatively easy to make, using lookup tables isn't new, lots of those languages are similar in syntax. The first time I tested the live API too it gave me a variable name "123456", so obviously they didn't think of the most obvious problems, rather than only keeping a list of keywords to check against.

Polymorphic encryption is meaningless without context too. I also mentioned cracking because I don't see there being any usefulness to this across a wide band of uses. It's a public service too.

Just being straight honest, believe what some of you kids will.

PELock - February 24, 2019, 6:56 am

"All of this would be relatively easy to make"

Didn't find any similar solution, except poor XOR codecs. But that's not the point. If you read the website you would know what's its purpose, I'm not going to repeat myself just because you are lazy.

" it gave me a variable name "123456""

You could just hit generate again, random things are random, you know?

Aristocles - February 25, 2019, 6:07 am

It reminds me of goold old

<https://zerosum0x0.blogspot.com/2017/08/obfuscatedencrypted-cc-online-string.html>

Nice stuff, however if its not free / does not include a free plan i'm gonna report the thread since it shall be in marketplace

have a nice day :-)

0xDEAD10CC - February 25, 2019, 8:52 am

(Feb 24, 2019 07:56 AM)PELock Wrote: [\[CITE\]](#) "All of this would be relatively easy to make"

Didn't find any similar solution, except poor XOR codecs. But that's not the point. If you read the website you would know what's its purpose, I'm not going to repeat myself just because you are lazy.

" it gave me a variable name "123456""

You could just hit generate again, random things are random, you know?
Well, the person above me found a very similar solution if you ask me.

The fact that you've found no similar solution doesn't mean this is by any means good. It also doesn't indicate that you even know how to search properly, or that you looked very hard. The website clearly says it is to prevent something like a basic hex editor from exposing the data. How is that not an attempt to protect against cracking/reverse-engineering?

"123456" was my very first try with the web service, which is pretty sad to begin with. As I said, the generated code is poor, there's conditions that the web service fails to check, and this is not going to prevent any good reverse engineer from dumping the data. Be as naive as you want about these facts, I wouldn't pay for this.

PELock - February 28, 2019, 4:43 am

Very similar solution? We are up since 2013, the solution you "found" is from 2017 and supports only 1 programming language, doesn't support UNICODE, doesn't support file encryption, doesn't have WebAPI, GUI clients.

I bet you wouldn't be able to crack it, because your analytic skills are way below checking basic facts like the dates.

You are troll.

Aristocles - February 28, 2019, 11:22 am

(Feb 28, 2019 05:43 AM)PELock Wrote: [\[CITE\]](#) Very similar solution? We are up since 2013, the solution you "found" is from 2017 and supports only 1 programming language, doesn't support UNICODE, doesn't support file encryption, doesn't have WebAPI, GUI clients.

I bet you wouldn't be able to crack it, because your analytic skills are way below checking basic facts like the dates.

You are troll.

Mister Wójcik, 0xDEAD10CC is also a reputable member & not only on this board, and I think all of this is a misunderstanding.

?

0xDEAD10CC - February 28, 2019, 6:31 pm

(Feb 28, 2019 05:43 AM)PELock Wrote: [CITE](#) Very similar solution? We are up since 2013, the solution you "found" is from 2017 and supports only 1 programming language, doesn't support UNICODE, doesn't support file encryption, doesn't have WebAPI, GUI clients.

I bet you wouldn't be able to crack it, because your analytic skills are way below checking basic facts like the dates.

You are troll.

Lol - Why would it not support unicode? ?

If you know how to deal with bytes, the decrypted bytes will still be the same of that of the unicode text bytes as input, even if you give it "shellcode" for the input string. It's up to the programmer using the string to use the string as though it is unicode and not just a single byte charset, otherwise the null terminator would end the string early. The fact that it supports shellcode means you can just insert the unicode bytes in UTF16-LE or BE; whatever you need... Explain how I can't use this for unicode text, please... I don't care if it only supports 1 language either, even psuedocode would be enough for me to translate it to any language I know in < 2 mins.

At least the blogger was smart enough to realize that it didn't need a web API or GUI clients. That's overkill and especially for something like this. And I can find issues and errors with your web service without even trying that hard.

404 right there.

Do you really think any good programmer is actually going to use or buy this? I'm just being blatantly straightforward here since nobody else is.

PELock - March 1, 2019, 2:58 pm

You found 404 error on non-existent page. And you found a raw JSON response code.

You are truly a hackerman :)

If you have to do everything manually - what's the point of it? Try my solution instead :)

(Feb 28, 2019 12:22 PM)Aristocles Wrote: [CITE](#) Mister Wójcik, 0xDEAD10CC is also a reputable member & not only on this board, and I think all of this is a misunderstanding.

?

Well, he bragged about other solutions invented years after mine.

He found 404 error on a non existent page and he thinks this is a bug.

He found out about raw JSON response codes and he thinks this is a bug.

You should really re-think about your view on this guy ;)

0xDEAD10CC - March 1, 2019, 7:50 pm

Are you retarded or do you just not even understand how your own website works? Your website was the one that provided me with the page containing the invalid link. I didn't pay to encrypt a file, so it fed back that raw ill-formed JSON in the body of the response. Even if your own website

doesn't work the way you think it should, that should say more than enough here... Why would the invalid link not be a bug? And why would anybody use a service by you if you can't even create a valid website?

The image clearly shows that you have no comprehension on how to properly escape the data.

You're also not helping your point here, since it definitely appears that a solution recently created is far better than yours even though it was made years ago. Put the pieces together - it doesn't take a scientist to figure out that you're a skid.

PELock - March 2, 2019, 2:07 pm

Raw JSON response is raw. Learn some basics first. Oh and learn about crypto, because it seems you don't know what polymorphic encryption is.

PS.

I got plenty of customers from here :-)

0xDEAD10CC - March 2, 2019, 3:01 pm

(Mar 2, 2019 03:07 PM)PELock Wrote: [CITE](#) Raw JSON response is raw. Learn some basics first. Oh and learn about crypto, because it seems you don't know what polymorphic encryption is.

PS.

I got plenty of customers from here :-)

I don't see how any of this is relevant to the issues I've brought up except for the fact that you either don't understand them, or are deliberately ignoring them because you don't know how to reply in an intelligent manner. There's no evidence to support any of your claims here. The facts of what polymorphic encryption is and isn't doesn't make this any less useless and buggy than it already is as a web service; aside from the fact that it isn't even related to the discussion here, and is likely just an attempt to defer people from thinking about your stupidity. I'm talking about all the issues in this web service, and you're talking about what you built it for.

The only reason why these are a web service is to protect your algorithms from being stolen I'd presume. However, on this topic code emulation or debugging can still easily discover anything in attempt to hide data using these mechanisms to any skilled individual in RE.

PELock - March 3, 2019, 3:40 am

One thing is for sure - you cannot crack it (maybe that's why you are so butthurt?).

The other thing is this thing is real, easy to use, UNICODE/ANSI is built-in, supports multiple languages, clients in PHP, online version, GUI version.

```
Code // encrypted with https://www.stringencrypt.com (v1.3.0) [C/C++]
// butthurt_user = "0xDEAD10CC"
wchar_t butthurt_user[11] = { 0x7C67, 0x5908, 0x1BA5, 0x1A44, 0x0EE3, 0x3D89, 0x6628,
0x64C7,
                                0x6B5F, 0x27FE, 0x1E9E };
```

```
for (unsigned int ORgyw = 0, jXuHp = 0; ORgyw < 11; ORgyw++)
{
    jXuHp = butthurt_user[ORgyw];
    jXuHp += ORgyw;
    jXuHp += 0x87C9;
    jXuHp = ((jXuHp << 11) | ( (jXuHp & 0xFFFF) >> 5)) & 0xFFFF;
```

```

jXuHp ^= 0x2038;
jXuHp -= ORgyw;
jXuHp ^= 0x01A0;
jXuHp += 0x2A8A;
jXuHp = ((jXuHp << 9) | (jXuHp & 0xFFFF) >> 7) & 0xFFFF;
jXuHp -= 0xC6D2;
jXuHp ^= 0xD62D;
jXuHp = ((jXuHp << 6) | (jXuHp & 0xFFFF) >> 10) & 0xFFFF;
jXuHp ^= 0x91DB;
jXuHp = (((jXuHp & 0xFFFF) >> 1) | (jXuHp << 15)) & 0xFFFF;
jXuHp -= 0x1718;
jXuHp ^= ORgyw;
jXuHp = (((jXuHp & 0xFFFF) >> 3) | (jXuHp << 13)) & 0xFFFF;
jXuHp ++;
jXuHp ^= ORgyw;
jXuHp = ~jXuHp;
butthurt_user[ORgyw] = jXuHp;
}

```

wprintf(butthurt_user);

0xDEAD10CC - March 3, 2019, 11:45 am

(Mar 3, 2019 04:40 AM)PELock Wrote: [\[CITE\]](#) One thing is for sure - you cannot crack it (maybe that's why you are so butthurt?).

The other thing is this thing is real, easy to use, UNICODE/ANSI is built-in, supports multiple languages, clients in PHP, online version, GUI version.

```

Code // encrypted with https://www.stringencrypt.com (v1.3.0) [C/C++]
// butthurt_user = "0xDEAD10CC"
wchar_t butthurt_user[11] = { 0x7C67, 0x5908, 0x1BA5, 0x1A44, 0x0EE3, 0x3D89, 0x6628,
0x64C7,
                                0x6B5F, 0x27FE, 0x1E9E };

```

```

for (unsigned int ORgyw = 0, jXuHp = 0; ORgyw < 11; ORgyw++)
{
    jXuHp = butthurt_user[ORgyw];
    jXuHp += ORgyw;
    jXuHp += 0x87C9;
    jXuHp = ((jXuHp << 11) | (jXuHp & 0xFFFF) >> 5) & 0xFFFF;
    jXuHp ^= 0x2038;
    jXuHp -= ORgyw;
    jXuHp ^= 0x01A0;
    jXuHp += 0x2A8A;
    jXuHp = ((jXuHp << 9) | (jXuHp & 0xFFFF) >> 7) & 0xFFFF;
    jXuHp -= 0xC6D2;
    jXuHp ^= 0xD62D;
    jXuHp = ((jXuHp << 6) | (jXuHp & 0xFFFF) >> 10) & 0xFFFF;
    jXuHp ^= 0x91DB;
    jXuHp = (((jXuHp & 0xFFFF) >> 1) | (jXuHp << 15)) & 0xFFFF;
    jXuHp -= 0x1718;
    jXuHp ^= ORgyw;
    jXuHp = (((jXuHp & 0xFFFF) >> 3) | (jXuHp << 13)) & 0xFFFF;
}

```

```
jXuHp ++;  
jXuHp ^= ORgyw;  
jXuHp = ~jXuHp;  
butthurt_user[ORgyw] = jXuHp;  
}
```

wprintf(butthurt_user);

What's so difficult about cracking an application with this as the only means of protection? I thought before you were implying that this wasn't intended to protect against cracking:

(Feb 22, 2019 07:44 PM)PELock Wrote: [\[CITE\]](#) 1. It's a polymorphic string encryption, not AES, RC4 or ECC, it's not meant to be anti-cracking solution (who said that?)

You're not only proving that what you say is one giant contradiction, but that you're also one incredibly naive member too... What's stopping somebody from breakpointing at that wprintf call?

PELock - March 3, 2019, 3:01 pm

If string encryption is an anti-cracking method to you - I congratulate you :D. Show us how you crack it Mr. Hackerman :)

It's the secret sauce of the polymorphism preventing only YOU from understanding what it is :)

0xDEAD10CC - March 3, 2019, 3:42 pm

(Mar 3, 2019 04:01 PM)PELock Wrote: [\[CITE\]](#) If string encryption is an anti-cracking method to you - I congratulate you :D. Show us how you crack it Mr. Hackerman :)

It's the secret sauce of the polymorphism preventing only YOU from understanding what it is :)

That's not even conceivably close to what it is. You've altered between calling it a protection mechanism and not being one at least 3 times throughout this thread and even on your website which entirely contradicts what you've been saying here; without the involvement of any of my posts. Clearly you're a moron.

Website: "String & file encryption will provide you just one layer of protection"

Your post: "it's not meant to be anti-cracking solution (who said that?)"

Your post: "you cannot crack it"

If anybody has issues with describing what it is and isn't, it's most obviously you. I'm not sure where your comprehension is falling short here, but all you need to do is re-read your posts if you still don't understand.

"String & File Encryption for VB.NET" - Provide an example binary using this for .NET and prove how this is useful by any means. Not everyone tries to find data using a hex editor.

PELock - March 4, 2019, 11:52 am

Can't do it huh?

This is source code encryption generator, what can't you understand? Heh ;)

Now crack this unhackable code obfuscation :P make my day, hack it, then crack it :)

0xDEAD10CC - March 5, 2019, 10:18 pm

(Mar 4, 2019 12:52 PM)PELock Wrote: [\[CITE\]](#) Can't do it huh?

This is source code encryption generator, what can't you understand? Heh ;)

Now crack this unhackable code obfuscation :P make my day, hack it, then crack it :)
This is like talking to a single dead brain cell... When you have any real experience and you can read, then maybe come back here, and read my post again.

PELock - March 6, 2019, 5:42 am

I'll come back here when you finally crack it like you've claimed before. Do your best because right now it's uncrackable ;) heh, so shut up and just do it.

0xDEAD10CC - March 7, 2019, 9:06 am

(Mar 6, 2019 06:42 AM)PELock Wrote: [\[CITE\]](#) I'll come back here when you finally crack it like you've claimed before. Do your best because right now it's uncrackable ;) heh, so shut up and just do it.
It's uncrackable because of your lacking comprehension skills. I requested a .NET binary that still doesn't exist because you avoided providing it.

Ajumma - March 7, 2019, 11:29 am

(Mar 6, 2019 06:42 AM)PELock Wrote: [\[CITE\]](#) I'll come back here when you finally crack it like you've claimed before. Do your best because right now it's uncrackable ;) heh, so shut up and just do it.
It's uncrackable because no one fucking understand what you're saying.
I thought you said this was not meant to be an anti-crack measure?
If you want people using your product and pay you MONEY, you gotta stop acting like a little butt-hurt kid when someone gives you criticism.

Aristocles - March 8, 2019, 12:37 am

I don't understand why people says its uncrackable anyway ? What could prevent me from BP on the MessageBox ? What could prevent me from Invoke() function ? What could prevent me from emulating the MSIL code in the method ? Or are you refering to the API itself that is uncrackable ?

I am reporting your drama anyway, thats not the place where to play i got the biggest one

fec-ri.muhaayel - March 8, 2019, 9:36 am

which encrypted strings ? in the epTlz ??

PELock - March 8, 2019, 9:50 am

0xDEAD10CC you can compile it yourself, it is source code ;)

The whole drama is about 0xDEAD10CC thinking the simple source code generator is an anti-cracking solution.

It's an anti-numnuts like you solution :)

PS. Also cures HIV :)

0xDEAD10CC - March 8, 2019, 7:24 pm

(Mar 8, 2019 10:50 AM)PELock Wrote: [\[CITE\]](#) 0xDEAD10CC you can compile it yourself, it is source code ;)

The whole drama is about 0xDEAD10CC thinking the simple source code generator is an anti-cracking solution.

It's an anti-numnuts like you solution :)

PS. Also cures HIV :)

Have you not read your own website?

PELock - March 9, 2019, 5:28 am

Nah, I'm just posting random terms just like you post random non-existent facts about it.

0xDEAD10CC - March 9, 2019, 12:39 pm

(Mar 9, 2019 06:28 AM)PELock Wrote: [\[CITE\]](#) Nah, I'm just posting random terms just like you post random non-existent facts about it.

Maybe you're posting them because what you've been saying has always been random and non-fact. I've been re-iterating everything that is mentioned on your website, which you're still too clueless to realize (even after I mention this).

There's no benefit to this, and you've failed to prove otherwise multiple times.

devvariety - March 10, 2019, 10:21 am

Using this myself, works awesome.

@0xDEAD10CC I don't see where on the site where this product claims anti-cracking, I believe you're looking at the sidebar where .netshrink advertises it.

0xDEAD10CC - March 10, 2019, 11:33 am

(Mar 10, 2019 10:21 AM)devvariety Wrote: [\[CITE\]](#) Using this myself, works awesome.

@0xDEAD10CC I don't see where on the site where this product claims anti-cracking, I believe you're looking at the sidebar where .netshrink advertises it.

It claims to be a protection mechanism, which could imply some sort of anti-cracking. Nothing to do with avoiding signature detection or anything aside from that. Which begs the question, what are you using it for then? It was a serious question, and seems the "creator" can't even come up with a decent answer even though I've pointed towards the wording on the site many times. The API requires an internet connection so it isn't even a standalone solution; whether you're using one of the provided SDK's or the client GUI...

devvariety - March 10, 2019, 3:10 pm

(Mar 10, 2019 11:33 AM)0xDEAD10CC Wrote: [\[CITE\]](#) It claims to be a protection mechanism, which could imply some sort of anti-cracking. Nothing to do with avoiding signature detection or anything aside from that. Which begs the question, what are you using it for then? It was a serious question, and seems the "creator" can't even come up with a decent answer even though I've pointed towards the wording on the site many times. The API requires an internet connection so it isn't even a standalone solution; whether you're using one of the provided SDK's or the client GUI...

Fair enough point, but overall I like it.

I use it to verify the authenticity of my server and client (vice versa). Without the proper calculations the client won't be able to tell if the server is authentic and vice versa.

0xDEAD10CC - March 10, 2019, 3:27 pm

(Mar 10, 2019 03:10 PM)devvariety Wrote: [\[CITE\]](#) Fair enough point, but overall I like it.

I use it to verify the authenticity of my server and client (vice versa). Without the proper calculations the client won't be able to tell if the server is authentic and vice versa.

I'd imagine this is some layer you're attempting to implement over top of SSL certificate validation?

devvariety - March 10, 2019, 3:35 pm

(Mar 10, 2019 03:27 PM)0xDEAD10CC Wrote: [\[CITE\]](#) I'd imagine this is some layer you're attempting to implement over top of SSL certificate validation?

I'm not using a SSL certificate, but my server is generating a keypair and sending the client the

public key, you know the rest. Since strings aren't safe in .NET memory, it's only good enough for those who don't know how to reverse engineer and that's good enough for me.

Aristocles - March 11, 2019, 4:30 am

@devvariety : this does not protect strings from memory, since the last emitting push put the decoded variable in a new local.

Quote:EncryptedString = EncryptedString + ChrW(BePuQ And &HFFFF)

btw the 'EncryptedString = EncryptedString' part is useless since the local at index EncryptedString is empty till this line

devvariety - March 11, 2019, 6:54 am

(Mar 11, 2019 04:30 AM)Aristocles Wrote: [\[CITE\]](#) @devvariety : this does not protect strings from memory, since the last emitting push put the decoded variable in a new local.

btw the 'EncryptedString = EncryptedString' part is useless since the local at index EncryptedString is empty till this line

I'm not using this to protect them from memory, only for authenticity. Without the correct calculations, again, you cannot reproduce the same string nor can you just send the UTF-8 string and hope for it to work, you need the correct calculations.

This is more than affordable guys, 3 string encrypts for \$1 (33 cents EURO) + tax, I love it.

0xDEAD10CC - March 11, 2019, 10:12 am

(Mar 11, 2019 06:54 AM)devvariety Wrote: [\[CITE\]](#) I'm not using this to protect them from memory, only for authenticity. Without the correct calculations, again, you cannot reproduce the same string nor can you just send the UTF-8 string and hope for it to work, you need the correct calculations.

This is more than affordable guys, 3 string encrypts for \$1 (33 cents EURO) + tax, I love it.

\$1 for authenticity? I still don't see the benefit of wasting a dollar. The "correct" calculations are still in the binary itself.

devvariety - March 11, 2019, 3:53 pm

(Mar 11, 2019 10:12 AM)0xDEAD10CC Wrote: [\[CITE\]](#) \$1 for authenticity? I still don't see the benefit of wasting a dollar. The "correct" calculations are still in the binary itself.

How would you get the correct calculation from the server itself if the client does not have the code to reproduce it?

Alright so lets say we're using C++ and you have a client and a server. The client wants to make sure that the server isn't an emulator so in the handshake process, the server has to send a string to the client and the client will verify it just by comparing strings. You use StringEncrypt on the string you're sending to the client.

The string being compared to anyone seeing the code in a decompiler (or w.e) is completely different. You can't just put the string you see and compare them because it won't compare with the string that server produced (if that makes any sense to anyone).

Same goes for client I guess.

0xDEAD10CC - March 11, 2019, 8:03 pm

(Mar 11, 2019 03:53 PM)devvariety Wrote: [\[CITE\]](#) How would you get the correct calculation from the

server itself if the client does not have the code to reproduce it?

Alright so lets say we're using C++ and you have a client and a server. The client wants to make sure that the server isn't an emulator so in the handshake process, the server has to send a string to the client and the client will verify it just by comparing strings. You use StringEncrypt on the string you're sending to the client.

The string being compared to anyone seeing the code in a decompiler (or w.e) is completely different. You can't just put the string you see and compare them because it won't compare with the string that server produced (if that makes any sense to anyone).

Same goes for client I guess.

Now that we're talking about encrypted bytes and decryption calculation separation -- If you're going to split up the encrypted data and the decoding mechanism between client/server, there's lots of standardized ways to protect the data itself on the client side. If you're hardcoding this information into your client, what's wrong with using standard encryption with a public/private key mechanism? This way the messages can also be validated. You should still be using SSL as a minimum for server authenticity validation btw, since if you aren't your server can probably easily be spoofed, and your "handshake" utilizing this StringEncrypt is pretty much useless.

devvariety - March 12, 2019, 9:05 am

I'm using Rsa already. I send the public key to the client and after the both sides agree on a shared secret, I use AES.

After client and server are ready, they start exchanging strings to make sure none of fake/emulators. All of this is done with a shared secret that was calculated under the process of using Rsa.

I find using SSL along with all of this is yes, more security and insurance, so I'll look into it. I just thought my setup was already secure, but thank you.

0xDEAD10CC - March 12, 2019, 9:13 am

(Mar 12, 2019 09:05 AM)devvariety Wrote: [\[CITE\]](#) I'm using Rsa already. I send the public key to the client and after the both sides agree on a shared secret, I use AES.

After client and server are ready, they start exchanging strings to make sure none of fake/emulators. All of this is done with a shared secret that was calculated under the process of using Rsa.

I find using SSL along with all of this is yes, more security and insurance, so I'll look into it. I just thought my setup was already secure, but thank you.

What you're describing here at least sounds secure, especially if you follow proper practice and use SSL for the traffic. What I'm finding hard to understand is where StringEncrypt benefits any of this if you're already using a public/private keypair and RSA?

devvariety - March 12, 2019, 9:21 am

(Mar 12, 2019 09:13 AM)0xDEAD10CC Wrote: [\[CITE\]](#) What you're describing here at least sounds secure, especially if you follow proper practice and use SSL for the traffic. What I'm finding hard to understand is where StringEncrypt benefits any of this if you're already using a public/private keypair and RSA?

Yes, it's secure. I just like to throw in some "jabs" so if anyone attempts to create an emulator for client or server. They would never be able to achieve it without the correct calculations for the strings.

"Dedicated" users would PROBABLY achieve it, but hey, anything can be reversed. I just like to prevent it as much as I can.

PELock - March 12, 2019, 1:27 pm

0xDEAD10CC can't swallow this pill heh, have you cracked it already? I demand some results!

BTW It's great for homomorphic encryption too, put that on your list :P

:D

devvariety - March 12, 2019, 1:36 pm

(Mar 12, 2019 01:27 PM)PELock Wrote: [CITE](#) 0xDEAD10CC can't swallow this pill heh, have you cracked it already? I demand some results!

BTW It's great for homomorphic encryption too, put that on your list :P

:D

I wouldn't underestimate him. Yes, I may like your product and support it but you can't underestimate the guy, he knows his stuff.

PELock - March 13, 2019, 7:11 am

Lets give a guy some credit for admitting its uncrackable :)