

Forum:[Hack Forums](#)

Thread: [JObfuscator Java Source Code Obfuscator](#)

PELock - July 28, 2021, 5:14 pm

JObfuscator is a source code obfuscator for the Java language. Protect Java source code & algorithms from hacking, cracking, reverse engineering, decompilation & technology theft.

<https://www.pelock.com/products/jobfuscator>

API for automation (PHP & Python)

<https://www.pelock.com/products/jobfuscator/api>

Windows & Linux clients

<https://www.pelock.com/products/jobfuscator/download>

Engine changes

v1.01 - 28.07.2021A new obfuscation strategy. For each method, extract all possible integers from the code and store them in an array. It makes the analysis harder because it requires an indexed table lookup for every numeric value.

v1.0 - 25.07.2021Initial release under the code name "Rome wasn't built in a day"

Change linear code execution flow to nonlinear version

Rename variable names to random string values

Rename method names to random string values

Shuffle order of methods in the output source

Encrypt strings using randomly generated polymorphic encryption algorithms

Client changes

v1.01 - 28.07.2021All clients (Windows, Linux) and SDKs (PHP & Python) updated to include the new obfuscation strategy /IntsToArrays.

v1.0 - 25.07.2021JObfuscator v1.0 main window

Initial release

Windows client

Command line version compatible with Windows and Linux (Mono)

WebAPI client with PHP usage examples

GenericSkid - July 29, 2021, 6:42 pm

Why are you obfuscating on the source level and not the bytecode level? Also its really weird that its a fully online service but still bundled as a client executable that sends a bunch of API POST requests. Could just be a web service at this point tbh.

PELock - July 31, 2021, 12:42 pm

Engine history

v1.02 - 30.07.2021

An integers to arrays obfuscation strategy, converts the integer values into double values to avoid deobfuscation by popular Java decompilers e.g. `double[] var_2597 = new double[]{13.898355719807563D, 65535.73657403742D, ... };`.

(Jul 29, 2021 06:42 PM)GenericSkid Wrote: [\[CITE\]](#) Why are you obfuscating on the source level and not the bytecode level?

Because that's how I design it. Most if not all Java obfuscators are bytecode obfuscators, I wanted to do something different and I'm really into source code processing and modification. Bytecode and source obfuscations gives you different opportunities

(Jul 29, 2021 06:42 PM)GenericSkid Wrote: [\[CITE\]](#) Also its really weird that its a fully online service but still bundled as a client executable that sends a bunch of API POST requests. Could just be a web service at this point tbh.

It's called SaaS (Software as a Service) and those requests are (a single one to be precise) Web API calls into our fully documented API interface

<https://www.pelock.com/products/jobfuscator/api>

More clients for the Web API - the better for the customers.

GenericSkid - July 31, 2021, 2:09 pm

(Jul 31, 2021 12:42 PM)PELock Wrote: [\[CITE\]](#) Because that's how I design it. Most if not all Java obfuscators are bytecode obfuscators, I wanted to do something different and I'm really into source code processing and modification.

Respectable point. Most of the AST parsing libraries I've seen have limitations that would make it hard to compete with the capabilities of a bytecode level obfuscator. If you roll your own that's an impressive amount of work.

PELock - July 31, 2021, 3:13 pm

(Jul 31, 2021 02:09 PM)GenericSkid Wrote: [\[CITE\]](#) Respectable point. Most of the AST parsing libraries I've seen have limitations that would make it hard to compete with the capabilities of a bytecode level obfuscator. If you roll your own that's an impressive amount of work.

Thank you. I know about those limitations, for example, JVM has a goto opcode, while there is no real goto in Java language. Most of the Java bytecode obfuscators repeat blindly all the known patterns in the obfuscation world like XORing constants, using goto opcode, I would love to add something unique to this genre and I slowly work on improvements. I think most obfuscators are missing obfuscations based on floating points calculations and that would be my goal to add new kinds of obfuscations.

More changes

Engine history

v1.03 - 31.07.2021

Some integers were not extracted correctly for the integers to arrays obfuscation strategy

Multilevel obfuscation of extracted integers into the arrays (into a random number of double and integer arrays)

Sample generated code <https://pastebin.com/b74t4T77>

PELock - August 5, 2021, 3:11 pm

Engine history

v1.04 - 05.08.2021

A new obfuscation strategy. Encrypt all integer values using more than 15 floating point math functions from the java.lang.Math.* class.

A new obfuscation strategy. For each method, extract all possible double values from the code and store them in an array.

Avoid renaming methods annotated with @Override keyword

String encryption obfuscation by default is using now more encryption instructions

Multi-pass execution for the integer and double extraction into arrays

Sample obfuscated output with floating-point obfuscations <https://pastebin.com/DxYRN7cZ>

PELock - August 17, 2021, 10:48 am

v1.07 - 17.08.2021

Improved integer to arrays obfuscation by generating multidimensional arrays e.g. `int[][] var_3035 = { { 65535 }, { 01, 02, 0b0 } };`

Improved doubles to arrays obfuscation by generating multidimensional arrays e.g. `double[][][] O_ZzBf7_4tcNvh_c = { { { 2.8 } }, { { 1.3 } }, { { 0.06, 3.7, 65535.8 } } };`

v1.06 - 16.08.2021

Improved non-linear code obfuscation by adding more randomness in it, e.g. random type of loops `while(...) {}` or `do {} while(...) {}`

v1.05 - 14.08.2021

Fix NaN numbers in integers encryption obfuscation

All generated integer type values are now randomly written in decimal, hexadecimal, binary or octal system e.g. `int[] mSjhhzs = { 010, 0x01BA7, 0x51, 1521, 0b1100 };`

PELock - October 19, 2021, 2:36 pm

v1.08 - 19.10.2021

Fixed a bug in methods renaming within the same class in our JObfuscator a Java source code #obfuscator

Free activation code 9D9A-81E5-2382-376C

<https://www.pelock.com/jobfuscator/>

yeah007 - November 11, 2021, 1:58 pm

I really need this for my java programs.

PELock - December 26, 2021, 3:43 pm

Thank you! Why not buy it and support its development?

PELock - August 24, 2022, 2:58 pm

Come play with out obfuscator and try to deobfuscate this code

<https://dev.to/bartosz/java-obfuscation-puzzle-1193>

The winning price is the free activation code!

MavenGradle - August 24, 2022, 3:06 pm

(Aug 24, 2022 02:58 PM)PELock Wrote: [\[CITE\]](#) Come play with out obfuscator and try to deobfuscate this code

<https://dev.to/bartosz/java-obfuscation-puzzle-1193>

The winning price is the free activation code!

```
Code    public static void main(String[] args)
    {
        double[] numArray = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
        double SD = calculateSD(numArray);
        System.out.format("Standard Deviation = %.6f", SD);
    }
    public static double calculateSD(double numArray[])
    {
        double sum = 0.0, standardDeviation = 0.0;
        int length = numArray.length;
        for(double num : numArray)
        {
            sum += num;
        }
        double mean = sum/length;
        for(double num: numArray)
        {
            standardDeviation += Math.pow(num - mean, 2);
        }
        return Math.sqrt(standardDeviation/length);
    }
}
```

PELock - August 24, 2022, 3:45 pm

The original source code was listed here <https://www.pelock.com/products/jobfuscator>

I know about that. You won the jerk award of the month, enjoy your cleverness :D

MavenGradle - August 24, 2022, 3:57 pm

(Aug 24, 2022 03:45 PM)PELock Wrote: [\[CITE\]](#) The original source code was listed here <https://www.pelock.com/products/jobfuscator>

I know about that. You won the jerk award of the month, enjoy your cleverness :D

I guess you're breaking the HF rules by not giving away the license key then. It's a complete deobfuscation and perfectly readable reply with the exact code / answer you were looking for.

Anyway. If you want to make the challenge more interesting, seeing how I restored the original code as you requested, why not write and then obfuscate something different than what you listed on your site?

I could also just do a Sysoutprintln with the calculation hardcoded as a string if it makes you feel any better?

Ps, the rules regarding giveaways:

<https://hackforums.net/allhelp.php>

Quote:Contests past their due time i.e. fake;

PELock - August 25, 2022, 2:43 am

You didn't deobfuscate shit, how did you come up with the original names of the variables if they

were completely removed? I tell you how - you did take it from my website. So technically you didn't deobfuscate it, just copied the original source from my website (my mistake).

Your winning prize - the idiot of the year award :D

Now cry me a f***** river!

MavenGradle - August 25, 2022, 2:55 am

(Aug 25, 2022 02:43 AM)PELock Wrote: [\[CITE\]](#) You didn't deobfuscate shit, how did you come up with the original names of the variables if they were completely removed? I tell you how - you did take it from my website. So technically you didn't deobfuscate it, just copied the original source from my website (my mistake).

Your winning prize - the idiot of the year award :D

Now cry me a f***** river!

Like I said: Make it harder. You're so predictable that I can get the original code with variables and everything using a single brain cell and 1 minute of work. Makes me wonder how good your products really are when you come up with shit like this. Anyone else feel the same way or is it just me?

That's all it took. A single brain cell to get the original source code back. So fix your mistake. And stop calling people idiots when the real idiot is you for posting a simple challenge like this and maybe stop crying about donations ([AsmJit](#) anyone ?) too, it's sad.

You're giving your own products a bad name with shit like this and I'm seriously disappointed in you for your marketing. PR is obviously not your strong suit.

In light of this, please watch this beginner's guide:

<https://www.youtube.com/watch?v=6bcviGzi84c>

PELock - August 25, 2022, 4:11 am

What was the goal?

To deobfuscate the code.

What you did? Took the original source from my site.

How smart of you. Did you do the deobfuscation? No.

Now you cry like a little bitch because you thought you can outsmart everyone while in fact you didn't do the task you did.

PS.

And if you read about Asmjit donations, I have donated to this project since beta version, I was the first to donate money and many times more afterwards, I had a deal with the author to include my link for the 1st ever donation made to it - and what did I get in return after the years have passed? A big fuck you for being the first person to support the project financially. And now, when they have plenty of other donations they forgot who supported it many times when they had zero customers... Petr Kobalicek is just an ungrateful bastard...

Screw him and screw you, how about that? Is that bad for a PR? Let me laugh even more :)

MavenGradle - August 25, 2022, 4:15 am

This is so sad. I guess you will never learn. Best of luck to you [PELock](#) . You should write more blog posts about it, you might end up on the github hall of cringe fails. That would be a great advertisement.

PELock - August 25, 2022, 1:44 pm

Stop it, get some help & learn how to deobfuscate shit NOT how to be a smartass.

I can get you 5% discount for your cleverness! I'm in awe of your googling skills.

WebDev - September 18, 2022, 5:42 am

i already know about it. thanks

Tizen - October 8, 2022, 9:51 am

(Aug 25, 2022 04:11 AM)PELock Wrote: [\[CITE\]](#) What was the goal?

To deobfuscate the code.

What you did? Took the original source from my site.

How smart of you. Did you do the deobfuscation? No.

Now you cry like a little bitch because you thought you can outsmart everyone while in fact you didn't do the task you did.

PS.

And if you read about Asmjit donations, I have donated to this project since beta version, I was the first to donate money and many times more afterwards, I had a deal with the author to include my link for the 1st ever donation made to it - and what did I get in return after the years have passed? A big fuck you for being the first person to support the project financially. And now, when they have plenty of other donations they forgot who supported it many times when they had zero customers... Petr Kobalicek is just an ungrateful bastard...

Screw him and screw you, how about that? Is that bad for a PR? Let me laugh even more :)
What a shame. I genuinely have a need for source-level obfuscation right now but I'll look elsewhere. Glad I read the comments. He won the giveaway fair and square, maybe next time separate your challenge from public domain.

PELock - October 11, 2022, 2:47 am

Of course, you "genuinely" do! I was crying all the way to the bank... hahahahahahahhahahahah

Bonjo - September 7, 2023, 3:54 am

Legit this is the worst obfuscator known to man

OWNER IS A RUDE POLISH CUNT, WHO IS SAD FAT AND A VIRGIN

PELock - November 9, 2023, 1:00 pm

But it's the most profitable software for me, I shit with shitcoins thanks to it! So get fucked!

PELock - May 13, 2026, 1:50 am

A new version of JObfuscator is available - <https://www.pelock.com/jobfuscator/> (desktop + cli
<https://www.pelock.com/products/jobfuscator/download>)

Engine changes

v1.30 - 14.05.2026 A new obfuscation strategy. Encrypt all double values using floating point math functions from the `java.lang.Math.*` class.

A new obfuscation strategy. Rebuild string literals as runtime-created String objects from hidden char arrays.

A new obfuscation strategy. Represent selected integer values through equivalent double math expressions.

A new obfuscation strategy. Inject opaque mixer chains and unreachable guards to increase the amount of code decompilers have to analyze.

A new obfuscation strategy. Replace plain true and false literals with equivalent runtime expressions.

A new obfuscation strategy. Wrap selected safe statements in noisy try/finally blocks while preserving program behavior.

A new obfuscation strategy. Encrypt int array contents and decode them at runtime.

A new obfuscation strategy. Encrypt double arrays by hiding their values as encoded bit patterns.

A new obfuscation strategy. Encrypt char arrays and restore the original characters through generated decoder code.

A new obfuscation strategy. Encrypt String array elements and rebuild them while the program runs.

A new obfuscation strategy. Split eligible string literals into nested `.concat(...)` chains so decompilers face more tokens and local data-flow steps; it runs before polymorphic string encryption when both are enabled.

Updated the StringEncrypt engine with support for nested polymorphic loops in string encryption.

Updated the engine parser configuration to support Java 21 source code constructs.

Client history

v1.3 - 14.05.2026 Windows and Linux clients updated with all the new obfuscation strategies

Windows client runs the obfuscation process in background now

Drag & drop support added for faster Java file opening

Command line version supports wildcards now and can process many files at once

All the SDK packages updated

Official Web API packages and repositories: Python (PyPI, GitHub), JavaScript (npm, GitHub), Rust (crates.io, GitHub), C# .NET (NuGet, GitHub), PHP Composer (Packagist, GitHub). Installation commands are on the API page.

Added more generous free demo options, that include 2000 char limit plus four obfuscation strategies

randomly rename methods

randomly rename variables

encrypt static `int[]` arrays initializers

expand strings into `char[]` arrays

Anyways, you can get a free code if you can read ;)

PELock - June 3, 2026, 3:44 pm

You can now fully automate and integrate obfuscation process by using JObfuscatore Maven pre-compiler plugin (you just need to upgrade your pom.xml configuration file).

<https://github.com/PELock/JObfuscatore-Maven-Compiler>

Engine changes

v1.40 - 03.06.2026 More guards added against Java source code limits for methods and statements (64 kB).

A new Self Check obfuscation strategy added to guard against deobfuscation (using reflection crc checks at runtime).

Client changes

v1.4 - 03.06.2026 Added JObfuscator Maven compiler plugin to automate obfuscation from any Maven supported workflow and IDE (just upgrade your pom.xml)

Added JObfuscator Client Maven plugin to use JObfuscator API calls

TonicBox - June 27, 2026, 1:10 pm

Neat project. string encryption and the int-extraction pass are real techniques, and writing your own engine definitely teaches you stuff. But tbh source-level is kind of a dead end for Java specifically.

The problem is javac gets the last word. Anything you emit has to survive being re-parsed and recompiled. So you're limited by java source spec and what JAVAC accepts at the source level. And worse, the compiler partially undoes your work on the way:

- constant folding
- DCE
- canonicalizing control flow
- And so on...

You limit yourself also from ever breaking decompilers or real analysis in any way (Irreducible flow, dynamic-family instruction nesting edge cases, type confusion, etc)

Bytecode obfuscators run after javac, so nothing gets a second pass at cleaning them up. Not saying you wasted your time, genuinely valuable if written in the interest of learning, there's places for complex source transformation stuff. Just, in practice, production application protection is not one of them.

PELock - July 1, 2026, 7:37 am

I have tested it against latest JEBDec with AI decompilation and it failed, so it's production ready.

Javac doesn't flatten control flow or optimize constant folding, because it's not $x = 1 + 2$ but its using external sources like arrays.

Bytecode obfuscation cannot achieve what I can do on AST high level and vice versa, I cannot use bytecode GOTO instruction for example.

So there are cons and pros.

Also - this isn't my last word, there are more improvements coming in.

Thank you for the comments, I might even build bytecode obfuscator to work along AST obfuscation to achieve even better results.

PELock - August 4, 2026, 3:02 am

v1.50 - 04.08.2026 Fixed method renaming (support for nested types and constructors).