

Forum:[Hack Forums](#)

Thread: [Unique Compiler Results \(Unique Signatures\)](#)

Velo - October 5, 2017, 4:45 am

So, I have a program that compiles source code from my text file into an executable.

I need EACH Compilation to be 100% unique/different

Currently I have random junk code, random variables/method names and SOME function randomization, etc; 50/50 chance of it using Thread.Sleep and Task.Delay, Application.Exit/Environment.Exit

What are some other good ways to enhance my signature uniqueness?

So far with sigbench im getting about 86% difference

Need this higher.

Thanks.

severos - October 6, 2017, 5:26 am

Group each method code into multiple 'blocks', for each block it should be safe to change the order of the lines in it.

Also, substitute the method call with the actual method implementation (this depends on the structure of the method).

I'm not really sure variables names would make difference, don't they compile to memory addresses after all? make sure you're building for 'Release' not 'Debug', 'Debug' contains much more info which could give false increase in the difference rate.

Finally, you can mess up with the result code and add some NOPs around.

Ninja - October 6, 2017, 7:29 am

(Oct 5, 2017 04:45 AM)Dota 2 Wrote: [\[CITE\]](#) 50/50 chance of it using Thread.Sleep and Task.Delay
I'm so fucking triggered.

Ranger. - October 6, 2017, 7:33 am

(Oct 5, 2017 04:45 AM)Dota 2 Wrote: [\[CITE\]](#) 50/50 chance of it using Thread.Sleep and Task.Delay
What about a 50/50 chance of it using Environment.Exit and Application.Exit?

gigajew - October 6, 2017, 7:59 am

or a 50/50 chance of memory leaks and copy pasta dripping out of the cd tray

GrandOIFlag - October 6, 2017, 10:43 am

(Oct 5, 2017 04:45 AM)Dota 2 Wrote: [\[CITE\]](#) So, I have a program that compiles source code from my text file into an executable.

I need EACH Compilation to be 100% unique/different

Currently I have random junk code, random variables/method names and SOME function randomization, etc; 50/50 chance of it using Thread.Sleep and Task.Delay

What are some other good ways to enhance my signature uniqueness?

So far with sigbench im getting about 86% difference

Need this higher.

Thanks.

thread.sleep and task.delay are two completely different things.

considering you're replacing things with things you've googled as alternatives i feel like this project isnt for you.

(Oct 6, 2017 07:33 AM)Ranger. Wrote: [\[CITE\]](#) What about a 50/50 chance of it using Environment.Exit and Application.Exit?
again, two completely different things.

Ranger. - October 6, 2017, 11:18 am

(Oct 6, 2017 10:43 AM)GrandOIFlag Wrote: [\[CITE\]](#) thread.sleep and task.delay are two completely different things.

considering you're replacing things with things you've googled as alternatives i feel like this project isnt for you.

again, two completely different things.

They're not completely different things, dumb-ass. They serve almost the same purpose (in each of the examples), which means that they're easily going to detected by any competent anti-virus. An anti-virus does not solely base a detection on a file-hash.

Stop trying so hard lmao.

GrandOIFlag - October 6, 2017, 11:22 am

(Oct 6, 2017 11:18 AM)Ranger. Wrote: [\[CITE\]](#) They're not completely different things, dumb-ass. They serve almost the same purpose (both of the examples), which means that they're easily going to detected by any competent anti-virus. An anti-virus does not solely base a detection on a file-hash.

Stop trying so hard lmao.

no they don't.

they achieve the same goal but differently.

Enviroment.Exit will kill the process disallowing you the opportunity to close and flush resources should only be used usually as last resort.

Application.Exit raises the OnClose event allowing you to dispose and take care of resources.

Ranger. - October 6, 2017, 11:24 am

(Oct 6, 2017 11:22 AM)GrandOIFlag Wrote: [\[CITE\]](#) they achieve the same goal

Exactly what I said. Stop trying to out-think this shit when you clearly know that I mean that they both END the process (Application.Exit might not end the process if there's a form preventing it from doing so though.)

GrandOIFlag - October 6, 2017, 11:29 am

(Oct 6, 2017 11:24 AM)Ranger. Wrote: [\[CITE\]](#) Exactly what I said. Stop trying to out-think this shit when you clearly know that I mean that they both END the process (Application.Exit might not end the process if there's a form preventing it from doing so though.)
that isn't the point here

task.delay and thread.sleep both cause a delay whether its async or sync but to say they're interchangeable is just ignorant.

THEY BOTH CAUSE A DELAY SO WE CAN 50/50 CHANCE IT SCREW THE REAL PURPOSE

VmX5 - October 6, 2017, 3:25 pm

(Oct 6, 2017 07:29 AM)Ninja Wrote: [\[CITE\]](#) I'm so fucking triggered.
LOL'd for like 5 minutes after reading this.

Velo - October 6, 2017, 5:48 pm

This is NOT for altering virus detection ratios.

clrlslwi - October 6, 2017, 6:09 pm

I think you're in way over your head if you actually have to ask this.. How about you develop your terrible p2c using your own research?

movsx - October 6, 2017, 8:10 pm

For each generated code to be 100% unique you need a metamorphic engine- see my tutorials series I've posted.

Golang - October 7, 2017, 7:41 pm

From the context I suggest you keep working out the infinite numbers of PI until you reach a number so long that it fills up the RAM and gives you a nice and sexy BSOD!

It will take dedication, but with the power of hope, anything is possible.

kevy42 - October 8, 2017, 1:55 pm

Well u cant get a 100% unique stub each time, there is always static or reappearing parts, unless ur obfuscating it like mad, wich is never good for anything..

what i do is i set reagions for each strip of code that often causes detections, than i Make the builder just choose what parts to use with a probabillity

Delirium™ - October 8, 2017, 6:40 pm

(Oct 6, 2017 11:24 AM)Ranger. Wrote: [\[CITE\]](#) Exactly what I said. Stop trying to out-think this shit when you clearly know that I mean that they both END the process (Application.Exit might not end the process if there's a form preventing it from doing so though.)

You're wrong and the guy you're quoting is actually correct. The fact that they do the same thing doesn't mean they're interchangeable. Task.Delay does not necessarily block the thread and can run asynchronously. Not to mention it can be cancelled. It also uses a timer behind the scene to do its job.

Just because you think they do the same thing, it doesn't necessarily mean you can swap one for the other and all is well.

gigajew - October 15, 2017, 9:26 pm

Instead of going about it this way, consider coding a built in obfuscator that takes care of the randomization