

Forum:[Hack Forums](#)

Thread: [NetCrack || Free MD5 Cracker || Use your clients to crack hashes](#)

BahNahNah - May 26, 2015, 10:27 pm

The more clients you have the quicker it will crack the hash.

The stub is .net 2.0 dependant.

This is using the [eSock class](#)

Screenshot:

Spoiler ([Click to View](#))

Stub output:

Spoiler ([Click to View](#))

[[IMG](#)]<http://www.refud.me/results/bqpbzexiumupovkgg7nugfhvgw0x0iqurl4.png>[[IMG](#)]

Download: ([Source](#)) ([Binary](#))

Only3 - May 26, 2015, 10:40 pm

(May 26, 2015 10:27 PM)BahNahNah Wrote: ([CITE](#)) The more clients you have the quicker it will crack the hash.

...

how its more quick i guess u send to all the clients the same chars so all do the same

BahNahNah - May 26, 2015, 10:43 pm

(May 26, 2015 10:40 PM)Only3 Wrote: ([CITE](#)) how its more quick i guess u send to all the clients the same chars so all do the same

Each client does a block. Its a bit hard to explain but each block is different, so clients arnt doing the same work as another client.

It could be quicker by sending a client multiple blocks, but i was too lazy.

Only3 - May 26, 2015, 10:53 pm

if so u have brain not that its new but good thing to code

BahNahNah - May 27, 2015, 12:38 am

(May 26, 2015 10:53 PM)Only3 Wrote: ([CITE](#)) if so u have brain not that its new but good thing to code

Thanks?

Spoiler ([Click to View](#))

Translate please

CodeAssassin - May 27, 2015, 1:34 am

(May 27, 2015 12:38 AM)BahNahNah Wrote: ([CITE](#)) Thanks?

Spoiler ([Click to View](#))

Translate please

I'm pretty sure he means even tho this is not a new idea at least you coded something of worth. Rather then the shit I have been outputting recently lol.

Good job mate this looks cool. Will Deffo check out your sauce :)

BahNahNah - May 27, 2015, 1:39 am

(May 27, 2015 01:34 AM)sweetysuckscock Wrote: [\[CITE\]](#) I'm pretty sure he means even tho this is not a new idea at least you coded something of worth. Rather then the shit I have been outputting recently lol.

Good job mate this looks cool. Will Deffo check out your sauce :)
Do you know a better way to keep the stub open?

My current method: (i know its bad)
Code while(true)
{
 Thread.Sleep(1000000);
}

CodeAssassin - May 27, 2015, 1:50 am

(May 27, 2015 01:39 AM)BahNahNah Wrote: [\[CITE\]](#) Do you know a better way to keep the stub open?

My current method: (i know its bad)
Code while(true)
{
 Thread.Sleep(1000000);
}How about a simple bool = false,

Set to true when the jobs complete and shut task after. Tbh I am on phone so can't check it out just yet. I'll get up download and have a look to see buddy :)

Can you not have a property(bool) in the connect class and the while true where it is empy(no thread.sleep) and just set to true when your operation is finished?

Apart from that would need to do some testing, This looks great btw, nice use of your esocks class :D

BahNahNah - May 27, 2015, 2:29 am

(May 27, 2015 01:50 AM)sweetysuckscock Wrote: [\[CITE\]](#) How about a simple bool = false,

Set to true when the jobs complete and shut task after. Tbh I am on phone so can't check it out just yet. I'll get up download and have a look to see buddy :)

Can you not have a property(bool) in the connect class and the while true where it is empy(no thread.sleep) and just set to true when your operation is finished?

Apart from that would need to do some testing, This looks great btw, nice use of your esocks class :D

The stub stays connected (or tries to be) to the server all the time.
Using just a while(true) loop would use too much cpu.

CodeAssassin - May 27, 2015, 2:53 am

(May 27, 2015 02:29 AM)BahNahNah Wrote: [\[CITE\]](#) The stub stays connected (or tries to be) to the server all the time.

Using just a while(true) loop would use too much cpu.

Hmm. What about in the while loop making it run some small operation constantly like. Get it to find it's own process in the taskmanager and just making it loop forever check if it's there in a using statement of course to keep the mem down. Else not really sure what you could run. I see your issue now you need it connected constantly.

BahNahNah - May 27, 2015, 3:06 am

(May 27, 2015 02:53 AM)sweetysuckscok Wrote: [\[CITE\]](#) Hmm. What about in the while loop making it run some small operation constantly like. Get it to find it's own process in the taskmanager and just making it loop forever check if it's there in a using statement of course to keep the mem down. Else not really sure what you could run. I see your issue now you need it connected constantly.

Wouldn't that also use high resources?

There must be a better way of doing it, how do rat stubs stay open?

CodeAssassin - May 27, 2015, 3:29 am

(May 27, 2015 03:06 AM)BahNahNah Wrote: [\[CITE\]](#) Wouldn't that also use high resources?

There must be a better way of doing it, how do rat stubs stay open?

Got one solution add a reference to Windows.forms and

```
Code System.Windows.Forms.Application.Run()
```

You're welcome :)

AlphaDelta - May 27, 2015, 3:43 am

(May 27, 2015 03:06 AM)BahNahNah Wrote: [\[CITE\]](#) Wouldn't that also use high resources?

There must be a better way of doing it, how do rat stubs stay open?

Choke the main thread, something like

PHP Code

```
<?php
```

```
while(running) Thread.Sleep(100);
```

should do just fine, the CPU usage is virtually non-existent.

(May 27, 2015 03:29 AM)sweetysuckscok Wrote: [\[CITE\]](#) Got one solution add a reference to Windows.forms and

```
Code System.Windows.Forms.Application.Run()
```

You're welcome :)

Application.Run(); is essentially (in this situation) equal to my previous code, it sets up a loop that checks for windows messages (purely anecdotal, this is just what I recall from my memory) and calls events.

CodeAssassin - May 27, 2015, 4:35 am

(May 27, 2015 03:43 AM)AlphaDelta Wrote: [\[CITE\]](#) Choke the main thread, something like

PHP Code

```
<?php
```

```
while(running) Thread.Sleep(100);
```

should do just fine, the CPU usage is virtually non-existent.

Application.Run(); is essentially (in this situation) equal to my previous code, it sets up a loop that checks for windows messages (purely anecdotal, this is just what I recall from my memory) and calls events.

Under the hood application run is a winapi function hooked into the message loop of the application, not quite a while loop more low lever then a normal while loop, So i expect it to preform a LOT better then while (true) thread.sleep

Code of message loop Application.Run() is hooked into

Code int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE, LPSTR lpCmdLine, int nCmdShow)

```
{
    MSG msg;
    BOOL bRet;

    while((bRet = GetMessage(&msg, NULL, 0, 0)) != 0)
    {
        if(bRet == -1)
        {
            // Handle Error
        }
        else
        {
            TranslateMessage(&msg);
            DispatchMessage(&msg);
        }
    }
    return msg.wParam;
}
```

From what i read the optimal way is to make the task a windows service and just have it run the tasks that need performing, could be an option.

d3agle - May 27, 2015, 4:51 am

(May 27, 2015 03:06 AM)BahNahNah Wrote: [CITE](#) Wouldn't that also use high resources?

There must be a better way of doing it, how do rat stubs stay open?

Take a look here to see how xRAT stays open.

<https://github.com/MaxXor/xRAT/blob/master/Client/Program.cs>

BahNahNah - May 27, 2015, 5:01 am

The stub now stays open with Application.Run and its now fud.

<http://www.refud.me/results.php?id=bqpbzexiumupovkgq7nugfhvgw0x0iqurl4>

morph - May 28, 2015, 6:25 am

Code while (true)

 Console.ReadKey()

or

 Thread.Sleep(Timeout.Infinite)

or

 Thread.Sleep(int.MaxValue)

bobolobo - May 28, 2015, 8:02 am

So i haven't time to look at source right now, since i am doing school stuff. But how efficient is this, how do you split the work efficiently in these "blocks" and compute a hash out of it without it taking up a lot of resources, especially later on?

I have done a similar project in the past (you can look it up), back then i had problems dividing the workload efficiently. It was without a doubt my biggest problem dividing the work and creating the string effectively without it taking up unnecessary resources, especially when having to compute a very large combination like the 10^9 th.

idb. - May 28, 2015, 12:36 pm

(May 27, 2015 02:29 AM)BahNahNah Wrote: [\[CITE\]](#) The stub stays connected (or tries to be) to the server all the time.

Using just a while(true) loop would use too much cpu.

ManualResetEvent class my friend. Learn it. Love it.

CodeAssassin - May 28, 2015, 1:45 pm

(May 28, 2015 12:36 PM)idb. Wrote: [\[CITE\]](#) ManualResetEvent class my friend. Learn it. Love it.

I just checked this out, that's a pretty cool class exactly what he needs :)

And great share idb, really cool never knew about his kind of threading where you can just basically put it on hold and wait until your ready for the other threads.

Thanks again :)

Wish i could give you some apples!

BahNahNah - May 28, 2015, 3:22 pm

(May 28, 2015 08:02 AM)bobolobo Wrote: [\[CITE\]](#) So i haven't time to look at source right now, since i am doing school stuff.

-snip-

A block is a 3 character string made up from the charset plus a suffix produced from the server.

In the GUI, you will see something like "???aa"

The 3 questionmarks are what the client produces and the readable characters is the suffix, so if your charset was "aAcC123" the client would hash and compare "AAA" and "333aa" and everything on between using the charset.

SIM-2233573 - May 29, 2015, 2:32 am

This is convenient. I'll vouch for this.

Hacklinx - May 30, 2015, 9:05 am

Very good idea, thank you i will see now the code.

Regards!