Attacking Common Applications Skills Assessment 3

Introduction

During our penetration test our team found a Windows host running on the network and the corresponding credentials for the Administrator. It is required that we connect to the host and find the `hardcoded password` for the MSSQL service.

target: 10.129.95.200

RDP to target with username "Administrator" and password "xcyj8izxNVzhf4z"

What is the hardcoded password for the database connection in the MultimasterAPI.dll file?

Connecting to the machine

```
xfreerdp3 /u:Administrator /p:xcyj8izxNVzhf4z /v:10.129.95.200
```

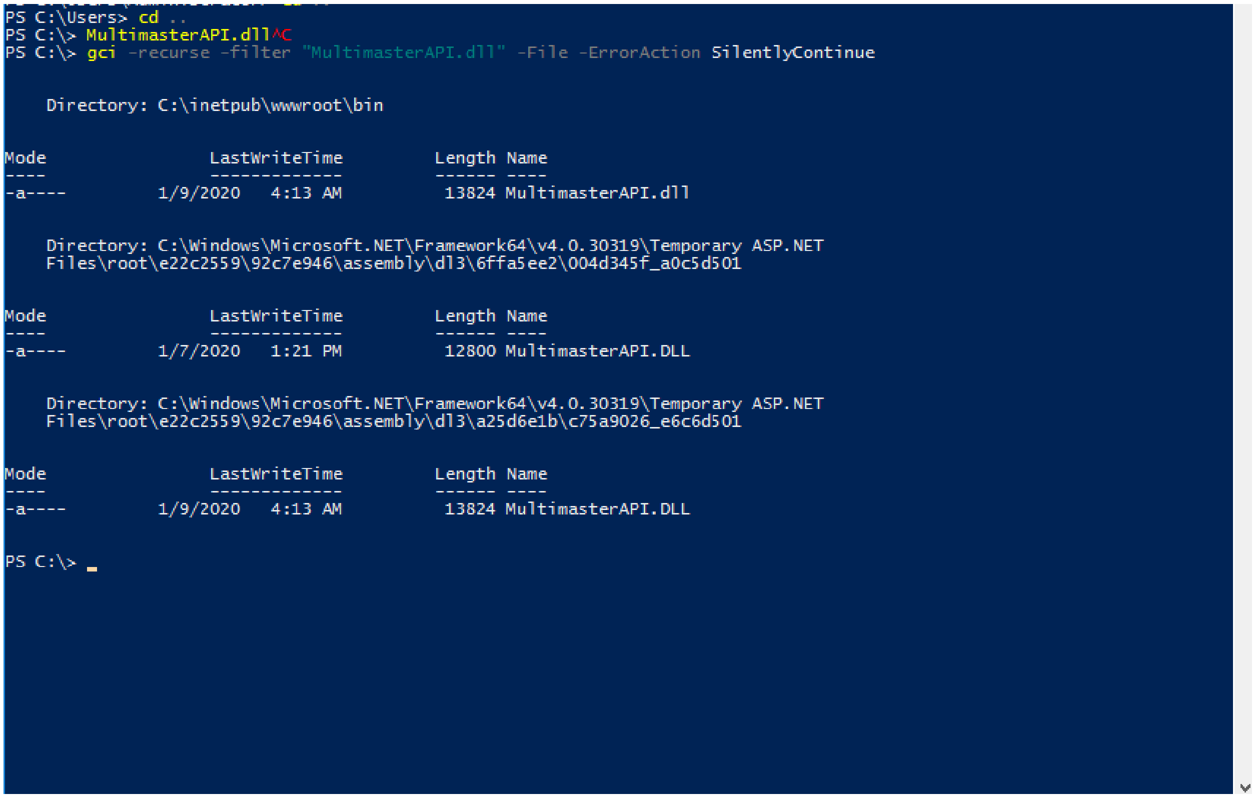
Finding the file theyre talking about

open powershell and navigate to the root C:\ (or you could specify this in the command using “

`-Path "C:\"`

```
the below command will recursively search a windows machine for a file:
```

```
gci -recurse -filter "MultimasterAPI.dll" -File -ErrorAction SilentlyContinue
```

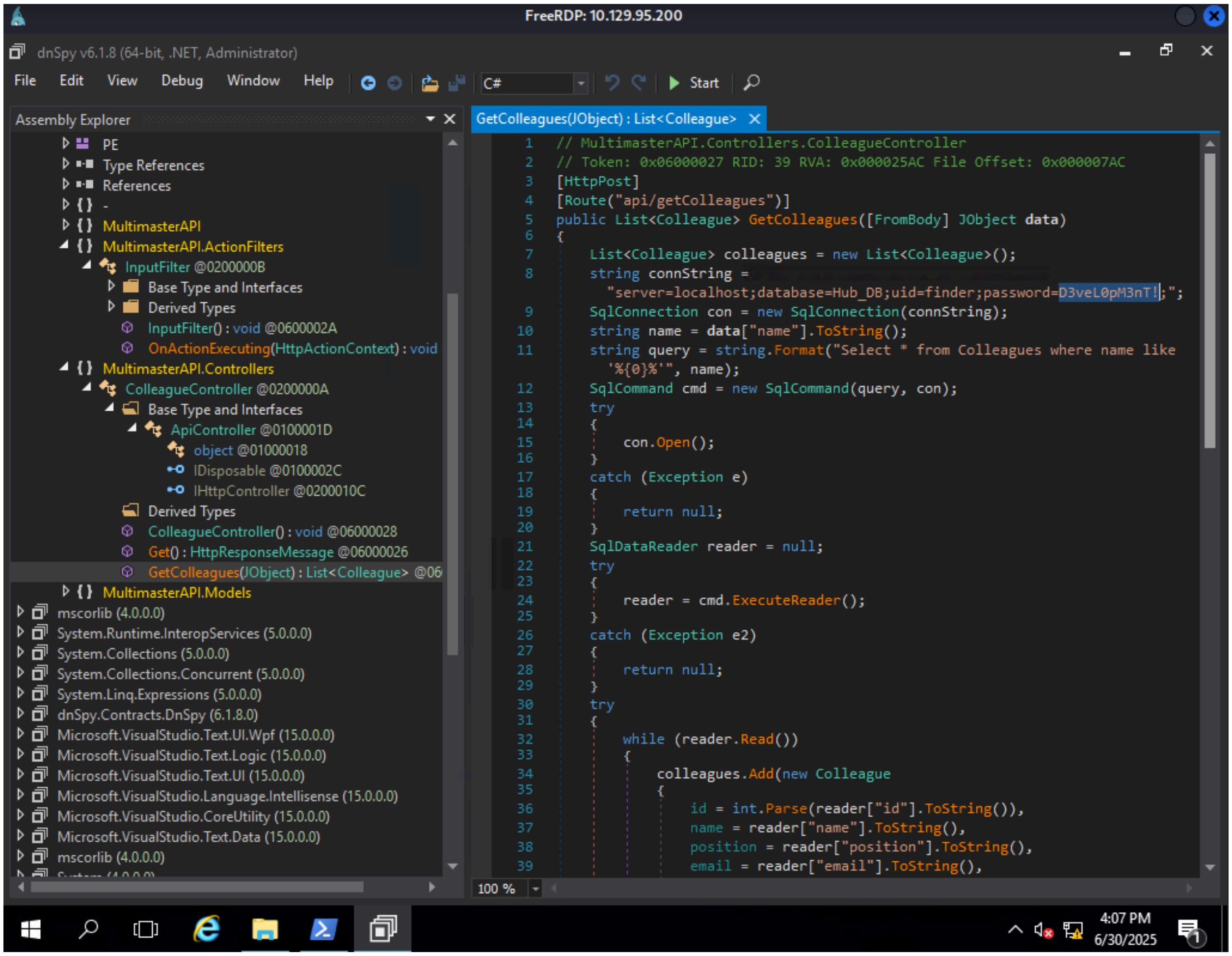


```
PS C:\Users> cd ..
PS C:\> MultimasterAPI.dll^C
PS C:\> gci -recurse -filter "MultimasterAPI.dll" -File -ErrorAction SilentlyContinue


    Directory: C:\inetpub\wwwroot\bin


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----         1/9/2020   4:13 AM          13824 MultimasterAPI.dll


    Directory: C:\Windows\Microsoft.NET\Framework64\v4.0.30319\Temporary ASP.NET
    Files\root\e22c2559\92c7e946\assembly\dl3\6ffa5ee2\004d345f_a0c5d501


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----         1/7/2020   1:21 PM          12800 MultimasterAPI.DLL


    Directory: C:\Windows\Microsoft.NET\Framework64\v4.0.30319\Temporary ASP.NET
    Files\root\e22c2559\92c7e946\assembly\dl3\a25d6e1b\c75a9026_e6c6d501


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----         1/9/2020   4:13 AM          13824 MultimasterAPI.DLL


PS C:\> _
```

based on the first one being in the webroot I think I will start my investigation into the files with that one.

Funnily enough when launching dnspy from the tools directory on the machine it opens up to MultimasterAPI.dll as the file so I guess I didn’t need to look for it

Clicking through the file to try and get an understanding of the file and what its doing I did find the hard coded credentials in a connection string.


FreeRDP: 10.129.95.200
dnSpy v6.1.8 (64-bit, .NET, Administrator)
File
Edit
View
Debug
Window
Help
C#
Start
Assembly Explorer
PE
Type References
References
{ } -
MultimasterAPI
MultimasterAPI.ActionFilters
InputFilter @0200000B
Base Type and Interfaces
Derived Types
InputFilter() : void @0600002A
OnActionExecuting(HttpActionContext) : void
MultimasterAPI.Controllers
ColleagueController @0200000A
Base Type and Interfaces
ApiController @0100001D
object @01000018
IDisposable @0100002C
IHttpController @0200010C
Derived Types
ColleagueController() : void @06000028
Get() : HttpResponseMessage @06000026
GetColleagues(JObject) : List<Colleague> @06
MultimasterAPI.Models
mscorlib (4.0.0.0)
System.Runtime.InteropServices (5.0.0.0)
System.Collections (5.0.0.0)
System.Collections.Concurrent (5.0.0.0)
System.Linq.Expressions (5.0.0.0)
dnSpy.Contracts.DnSpy (6.1.8.0)
Microsoft.VisualStudio.Text.UI.Wpf (15.0.0.0)
Microsoft.VisualStudio.Text.Logic (15.0.0.0)
Microsoft.VisualStudio.Text.UI (15.0.0.0)
Microsoft.VisualStudio.Language.Intellisense (15.0.0.0)
Microsoft.VisualStudio.CoreUtility (15.0.0.0)
Microsoft.VisualStudio.Text.Data (15.0.0.0)
mscorlib (4.0.0.0)
GetColleagues(JObject) : List<Colleague>
100 %
4:07 PM
6/30/2025

```csharp
// MultimasterAPI.Controllers.ColleagueController
// Token: 0x06000027 RID: 39 RVA: 0x000025AC File Offset: 0x000007AC
[HttpPost]
[Route("api/getColleagues")]
public List<Colleague> GetColleagues([FromBody] JObject data)
{
    List<Colleague> colleagues = new List<Colleague>();
    string connString =
      "server=localhost;database=Hub_DB;uid=finder;password=D3veL0pM3nT!;";
    SqlConnection con = new SqlConnection(connString);
    string name = data["name"].ToString();
    string query = string.Format("Select * from Colleagues where name like
      '%{0}%'", name);
    SqlCommand cmd = new SqlCommand(query, con);
    try
    {
        con.Open();
    }
    catch (Exception e)
    {
        return null;
    }
    SqlDataReader reader = null;
    try
    {
        reader = cmd.ExecuteReader();
    }
    catch (Exception e2)
    {
        return null;
    }
    try
    {
        while (reader.Read())
        {
            colleagues.Add(new Colleague
            {
                id = int.Parse(reader["id"].ToString()),
                name = reader["name"].ToString(),
                position = reader["position"].ToString(),
                email = reader["email"].ToString(),
```

Because I stumbled upon the answer, looking into things a bit more for it appears you can use ctrl+shift+k to search assemblies to parse the DLL for things that may be of interest. If the hard coded credentials happen to be specifically stored in a string you can filter for that too.