PMA 401. Simple EXE Hacking with Ollydbg

What You Need

A Windows 2016 Server machine, real or virtual. Other Windows versions should also work.

Purpose

To modify a Windows EXE file and save an altered version. This gives you practice with very simple features of the Ollydbg debugger.

Downloading OllyDbg

Download OllyDbg 1.10 here:

http://www.ollydbg.de/download.htm

Right-click the file and click **Extract**, "**Extract All...**".

Double-click the red icon to launch it.

Downloading HashCalc

Download HashCalc here:

https://www.slavasoft.com/hashcalc/

Unzip it and double-click the **setup** to install it. Install it with the default options.

Task 1: Target EXE Recon

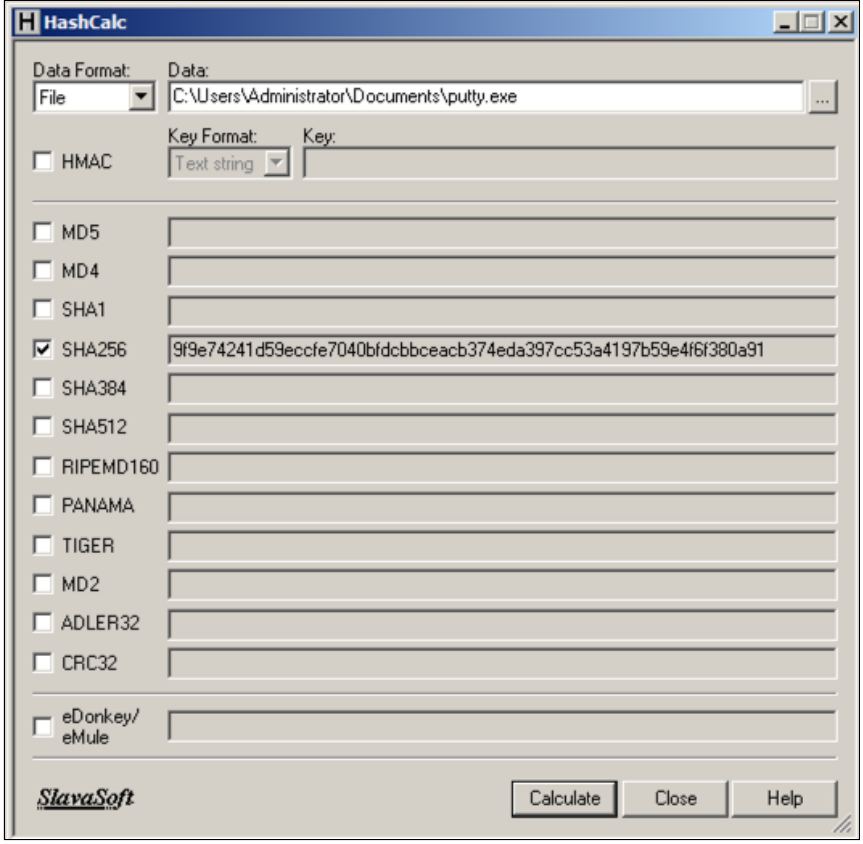
Get putty.exe

Download putty here:

https://samsclass.info/127/proj/putty.exe

Verifying the SHA256 Hash

Run Hashcalc on **putty.exe** and confirm that the SHA256 value matches the value shown below.



Running Putty

Double-click **putty.exe**. PuTTY opens, as shown below.

If PuTTY won't start, right-click it, click **Properties**, and click **Unblock**.

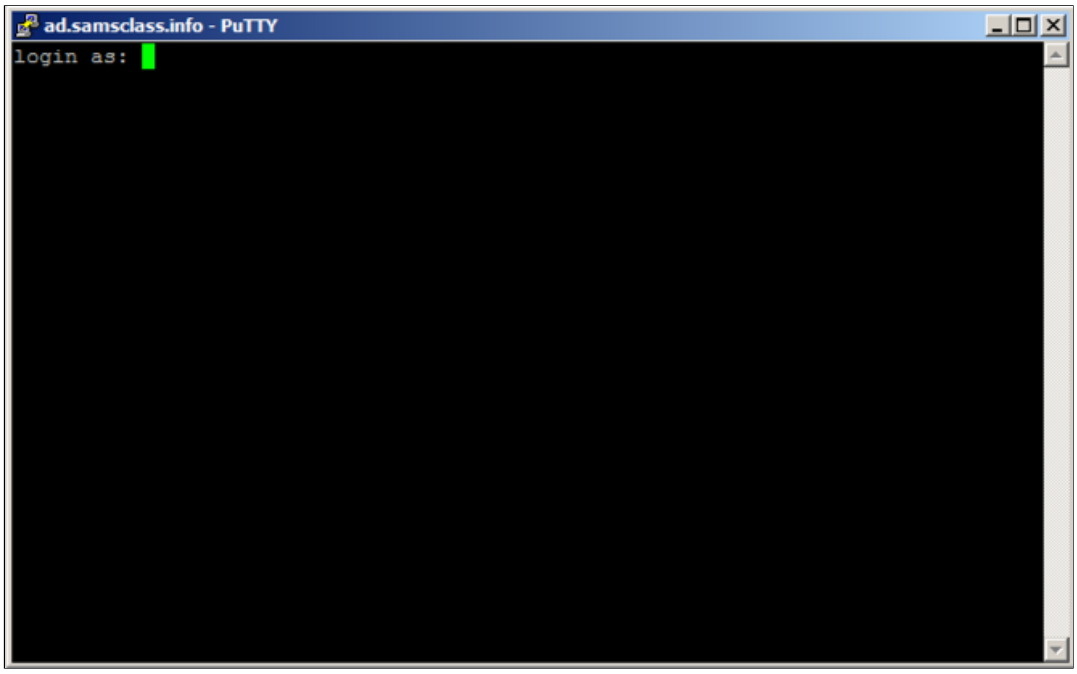
In the "Host Name (or IP address)" box, type

```
ad.samsclass.info
```

At the bottom, click the **Open** button.

If a "PuTTY Security Alert" box pops up, click **Yes**.

A black box opens, and shows a "**login as:**" prompt, as shown below.



You could connect to a server at this point, but that's not the point of this project. We will alter this program to do other things instead of printing "login as".

Close the Putty window.

Starting Ollydbg

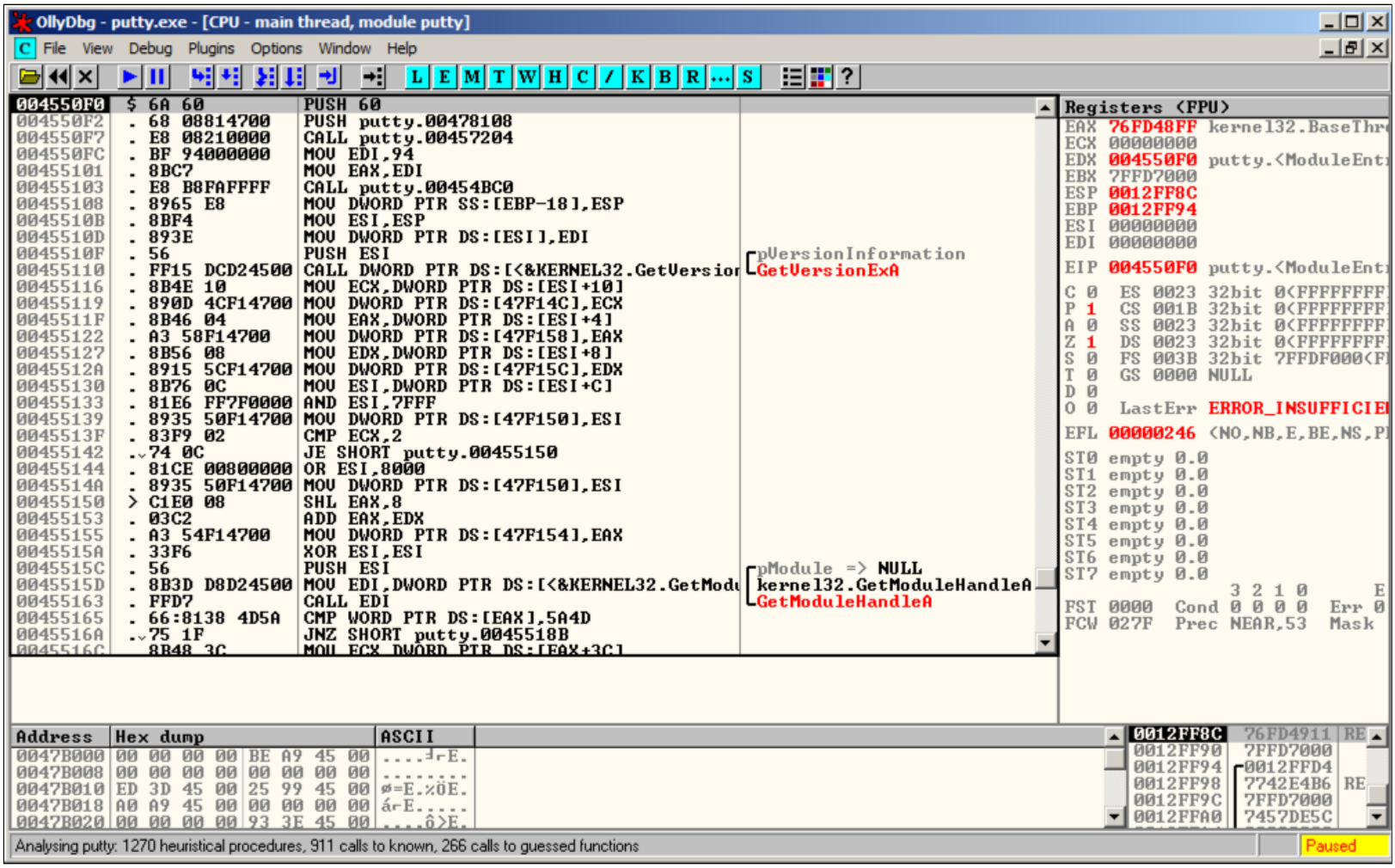
Launch OllyDbg.

If messages ask about deleting old DLL,s click **Yes**.

In Ollydbg, from the menu bar, click **File**, **Open**. Navigate to **putty.exe** and open it.

Ollydbg opens, as shown below. If your screen doesn't look like this, click **View**, **CPU** and maximize the CPU window.

(If you are using a 64-bit Windows system, the assembly code will look different from the image below until you execute the first Run command. Don't worry about that now.)



Ollydbg shows you a lot of data, but for now just notice the **Assembly Code** in the top left pane, and the **Paused** message in the lower right.

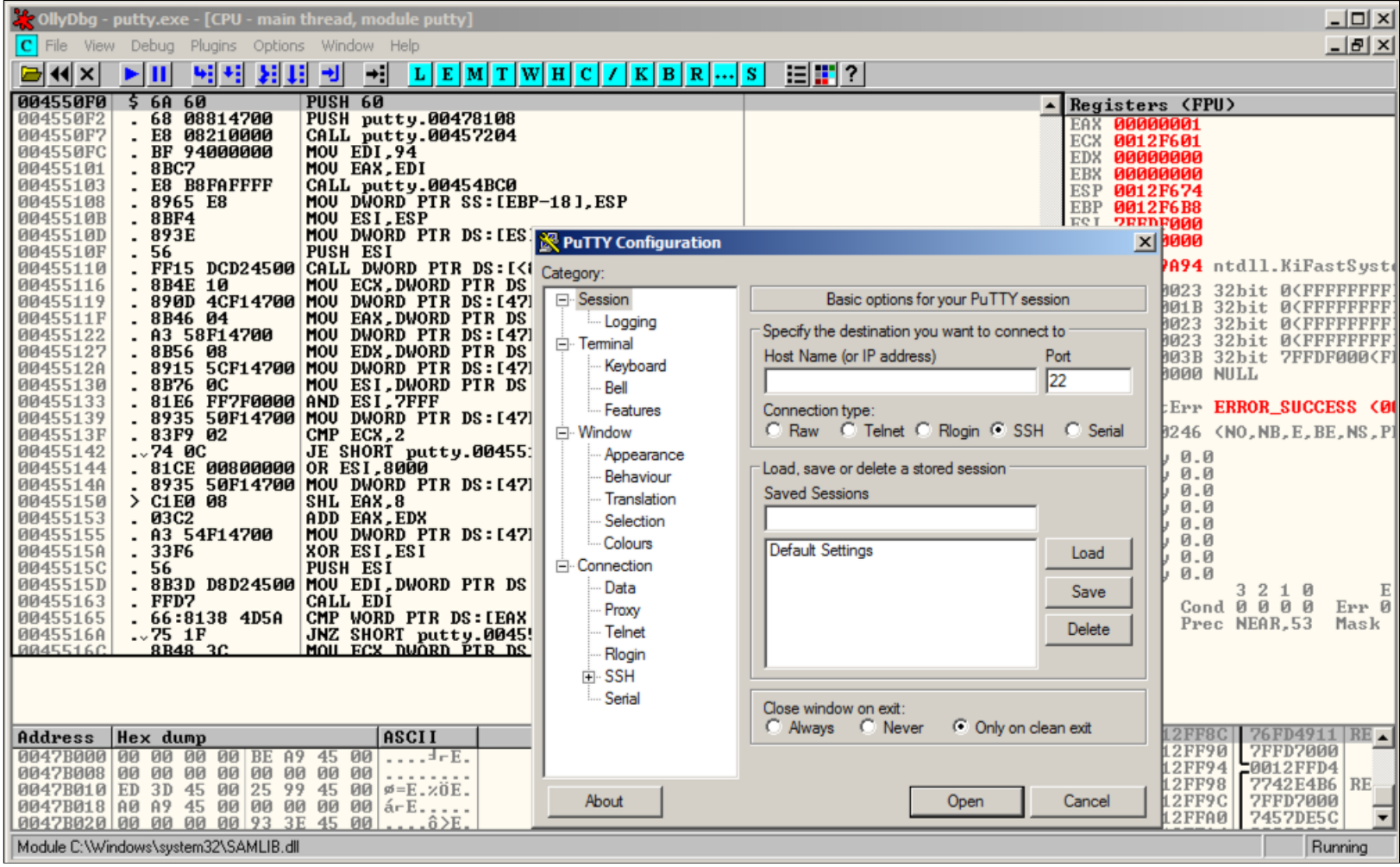
When you load a program into Ollydbg, it starts in a "Paused" state, with the Assembly Code window showing the first instruction.

Running Putty in Ollydbg

In Ollydbg, from the menu bar, click **Debug**, **Run**.

If the lower-right corner of OllyDbg still shows a "Paused" message, click **Debug**, **Run** again.

A Putty window opens, but it's behind the Olly window. At the bottom of the screen, in the taskbar, click the "**PuTTY Configuration**" button to bring the PuTTY window to the front, as shown below.

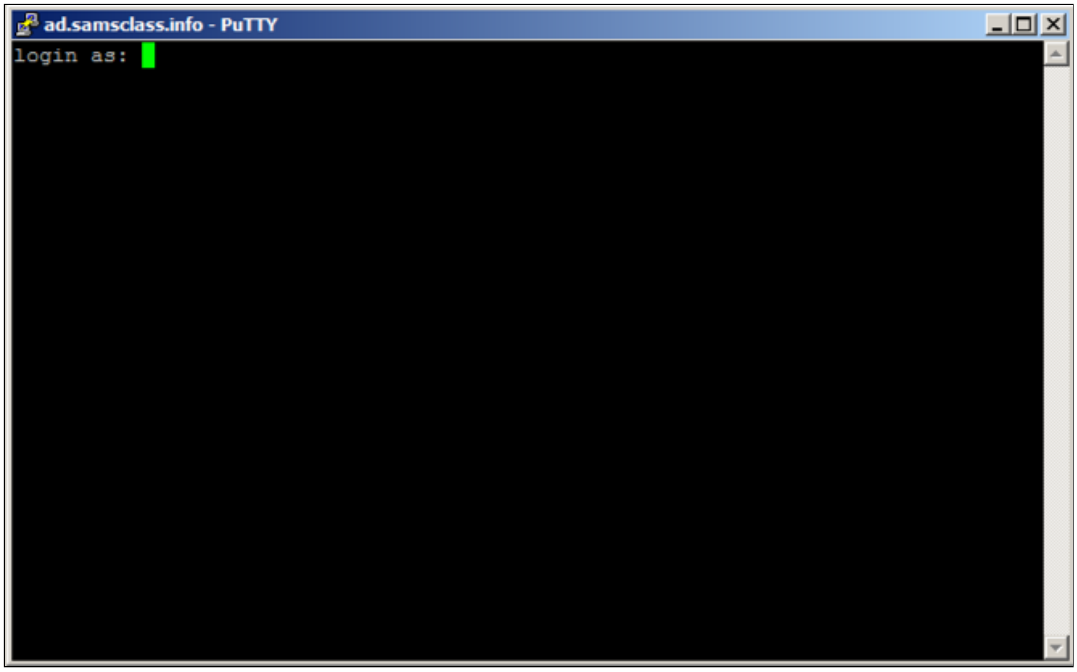


Click in the Putty window. In the "Host Name (or IP address)" box, type

```
ad.samsclass.info
```

At the bottom, click the **Open** button.

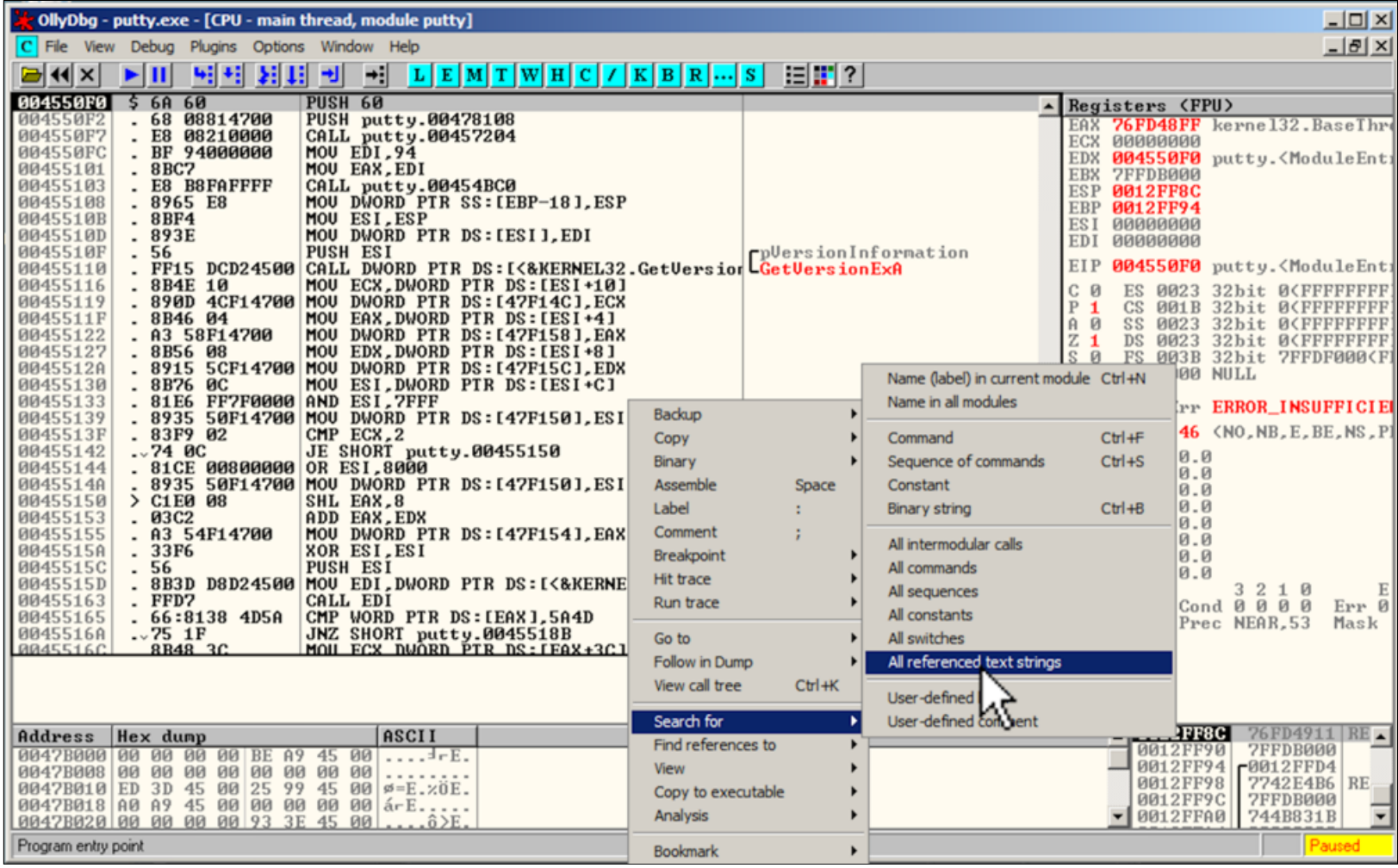
The "login as" message appears, as shown below.



Putty is running, but it's under the control of Ollydbg, so we can modify its execution.

Finding the "login as" Code

In Ollydbg, in the "Assembly Code" pane, right-click. Point to "**Search for**". Click "**All referenced text strings**", as shown below.



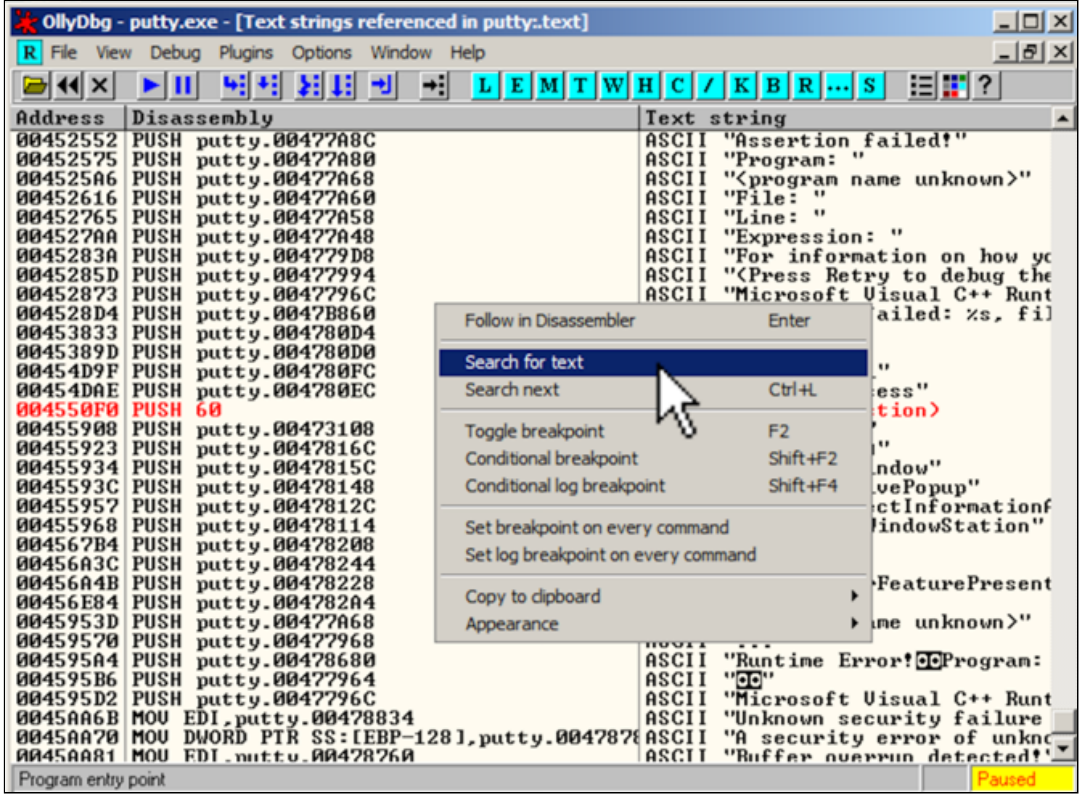
A "Text strings referenced in putty:.text" window opens, showing all the strings in the program.

Troubleshooting

If the title of this window does not contain "putty", but sone library such as "ntdll", click "Debug", "Restart" and run Putty again.

To make this text easier to read, right-click, point to **Appearance**, **Font**, and click "**OEM Fixed Font**".

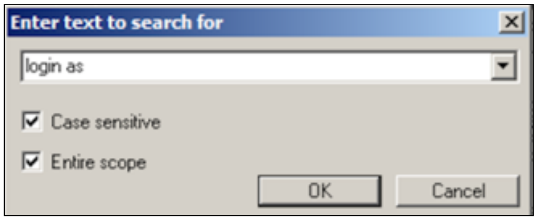
Right-click in that window, and click "**Search for text**", as shown below.



In the "Enter text to search for" box, type

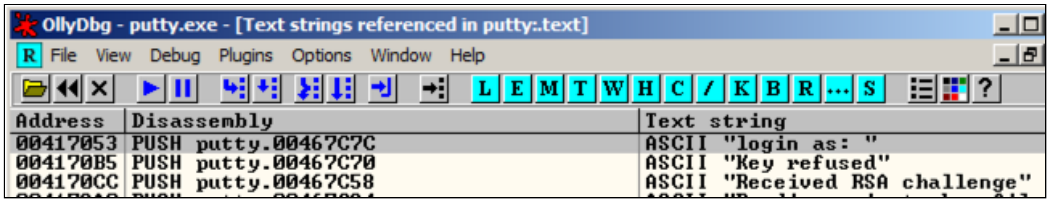
```
login as
```

as shown below. Check the "**Entire scope**" box.



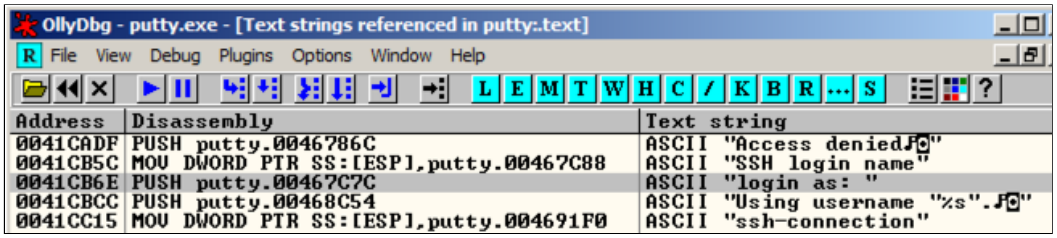
Click **OK**.

Ollydbg finds the ASCII string "login as", and the instruction that uses it, as shown below. This instruction is at address 00417053.



Right-click again, and click "**Search next**".

Ollydbg finds another line of code that uses this string, as shown below. This instruction is at address 0041CB6E.



Right-click again, and click "**Search next**".

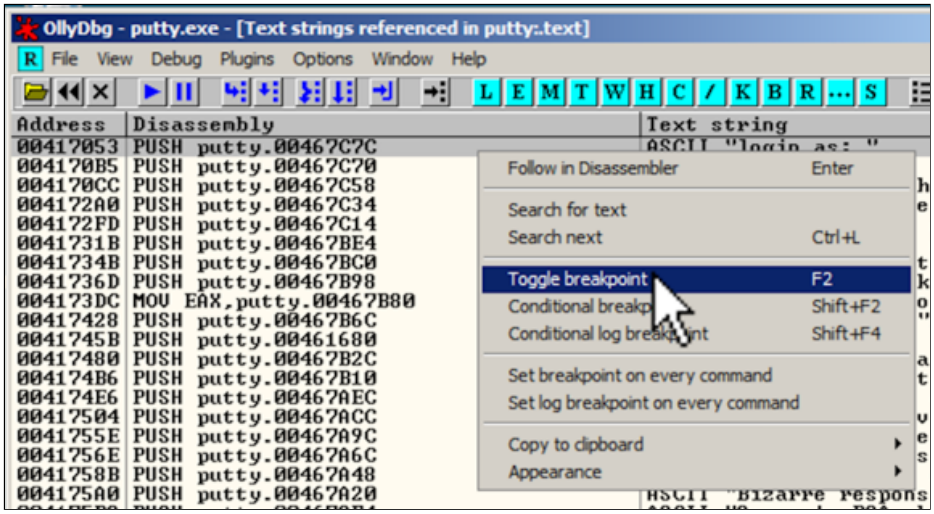
A message appears at the bottom of the window saying "Item not found". There are only two commands in the program that use this string.

Using Breakpoints

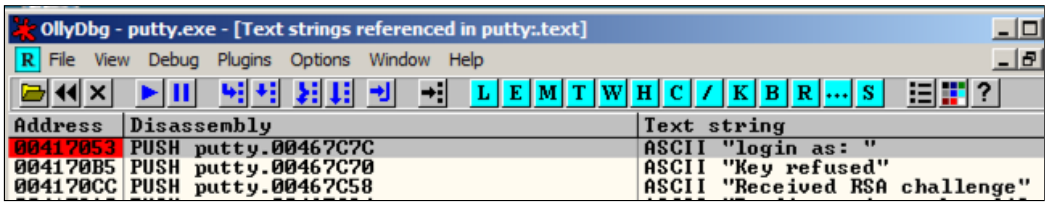
We'll set breakpoints at those instructions to see which one is used when logging in to an SSH server.

In the "Text strings referenced in putty:.text" window, right-click again, and click "**Search text**". In the "Enter text to search for" box, click **OK**.

The instruction at 00417053 appears again. Right-click this instruction and click "**Toggle breakpoint**", as shown below.

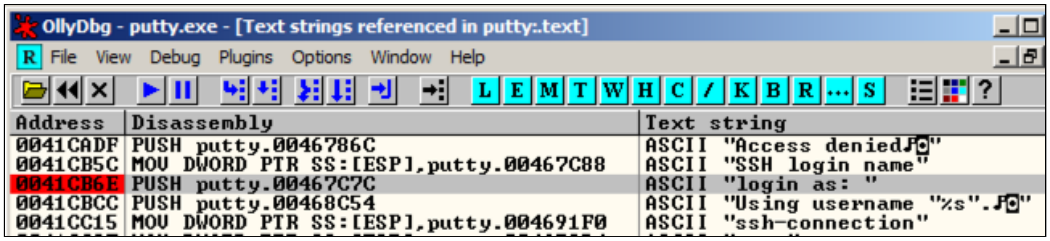


The address turns red, as shown below, to indicate that there's a breakpoint here.



Right-click again, and click "**Search next**". The instruction at address 0041CB6E appears. Right-click it and click "**Toggle breakpoint**".

The address turns red, as shown below.



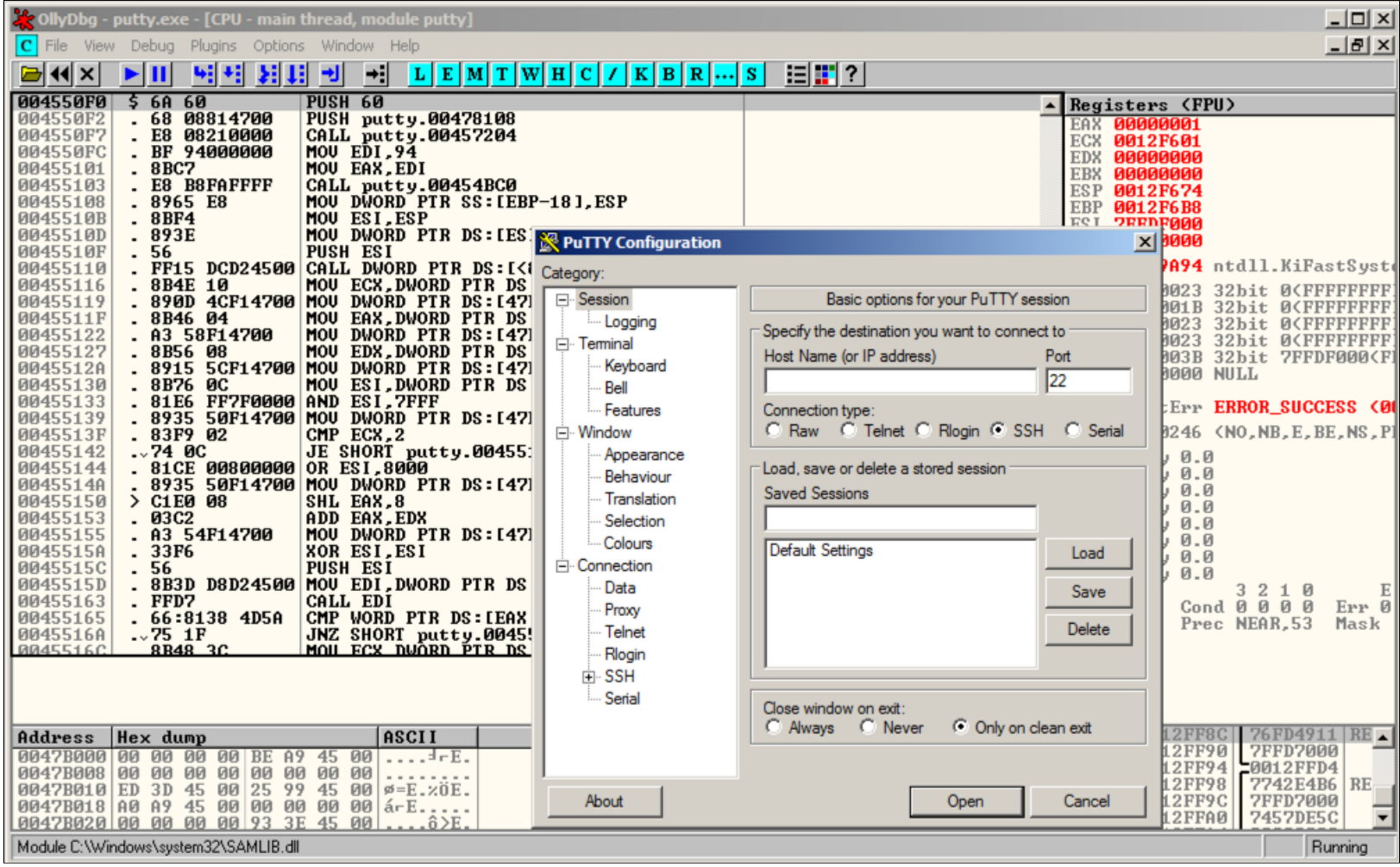
In Ollydbg, from the menu bar, click **Debug**, **Restart**.

A box pops up warning you that "Process 'putty' is active". Click **Yes**.

In Ollydbg, from the menu bar, click **Debug**, **Run**.

If the lower-right corner of OllyDbg still shows a "Paused" message, click **Debug**, **Run** again.

A Putty window opens. Bring it to the front, as shown below.

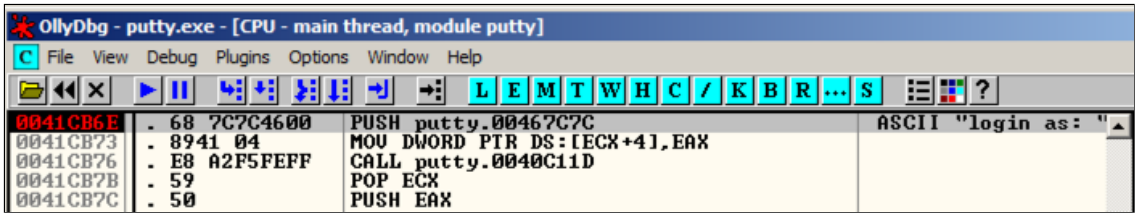


Click in the Putty window. In the "Host Name (or IP address)" box, type

```
ad.samsclass.info
```

At the bottom, click the **Open** button.

A black window opens and closes quickly, and the program stops, as shown below.



The program stopped at instruction 0041CB6E, as shown in the image above.

We'll use this instruction to hijack the program's execution.

Task 2: Alter the Login Message

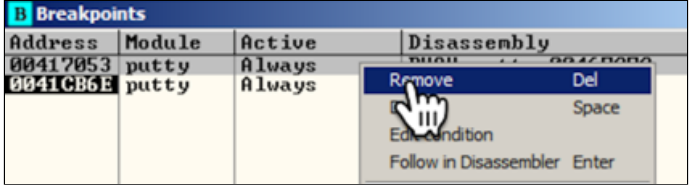
Removing the Breakpoints

We don't need the breakpoints any more, so we'll remove them.

In Ollydbg, from the menu bar, click **View**, **Breakpoints**.

A "Breakpoints" window opens, showing two breakpoints.

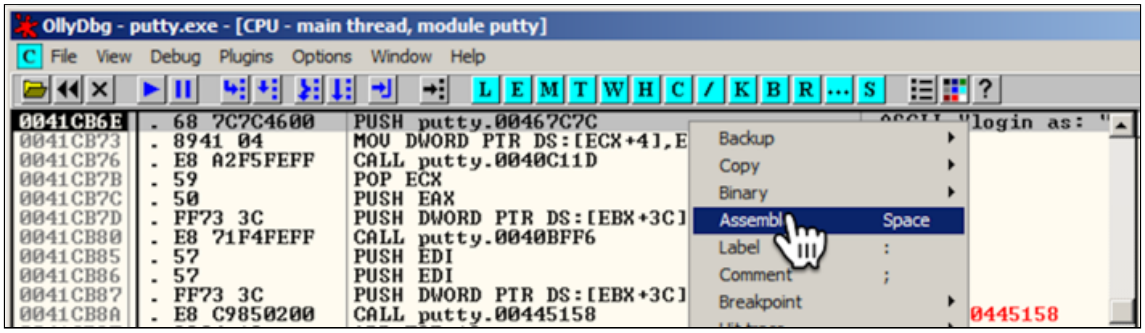
Right-click the first breakpoint and click **Remove**, as shown below.



Repeat the process to remove the other breakpoint. Close the "Breakpoints" window.

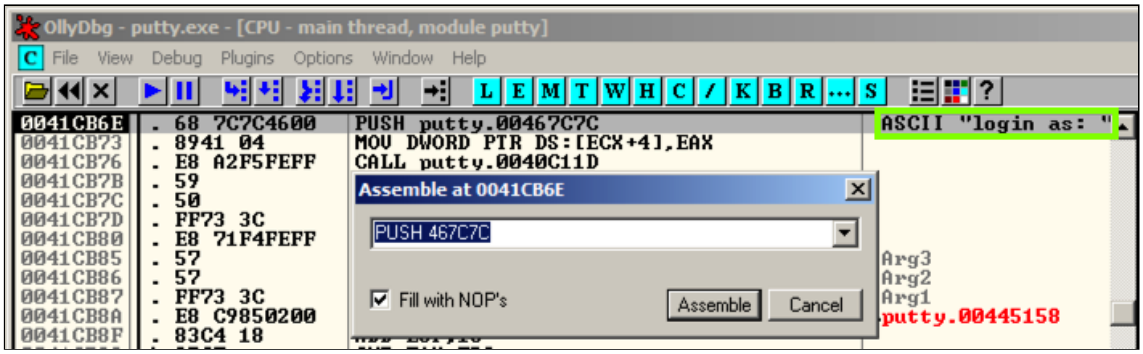
Removing One Letter From the Message

In Ollydbg, in the CPU window, in the Assembly Code pane, right-click the instruction at address **0041CB6E** and click **Assemble**, as shown below.

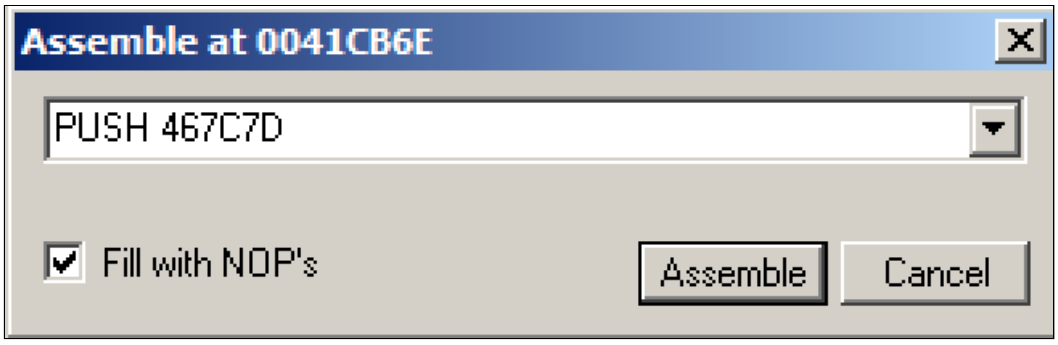


An "Assemble at 0041CB6E" box appears, as shown below.

This shows the command at this location. It's a PUSH instruction, placing the address 467C7C onto the stack. That address points to the letter "l" in the ASCII string "login as: ", as shown on the right side of the instruction line, outlined in green in the image below.



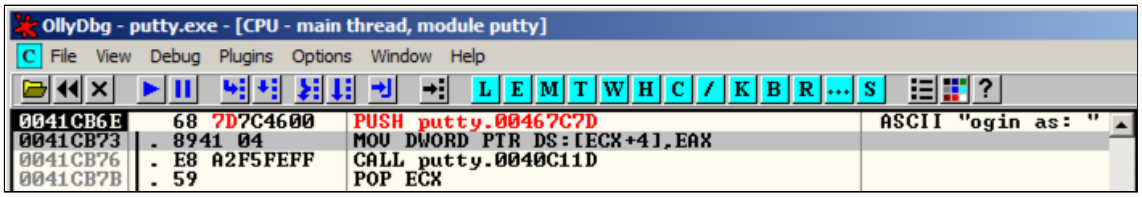
In the "Assemble at 0041CB6E" box, change the last character to **D**, as shown below. This will move the pointer from the "l" to the "o" in the string "login as: ".



Click the **Assemble** button.

Click the **Cancel** button.

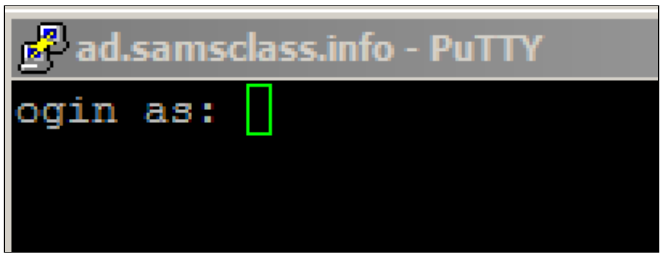
The message on the right now says "ogin as: ", as shown below.



Running the Modified Program

In Ollydbg, from the menu bar, click **Debug**, **Run**.

The black login window appears, with the message "ogin as: ", as shown below.

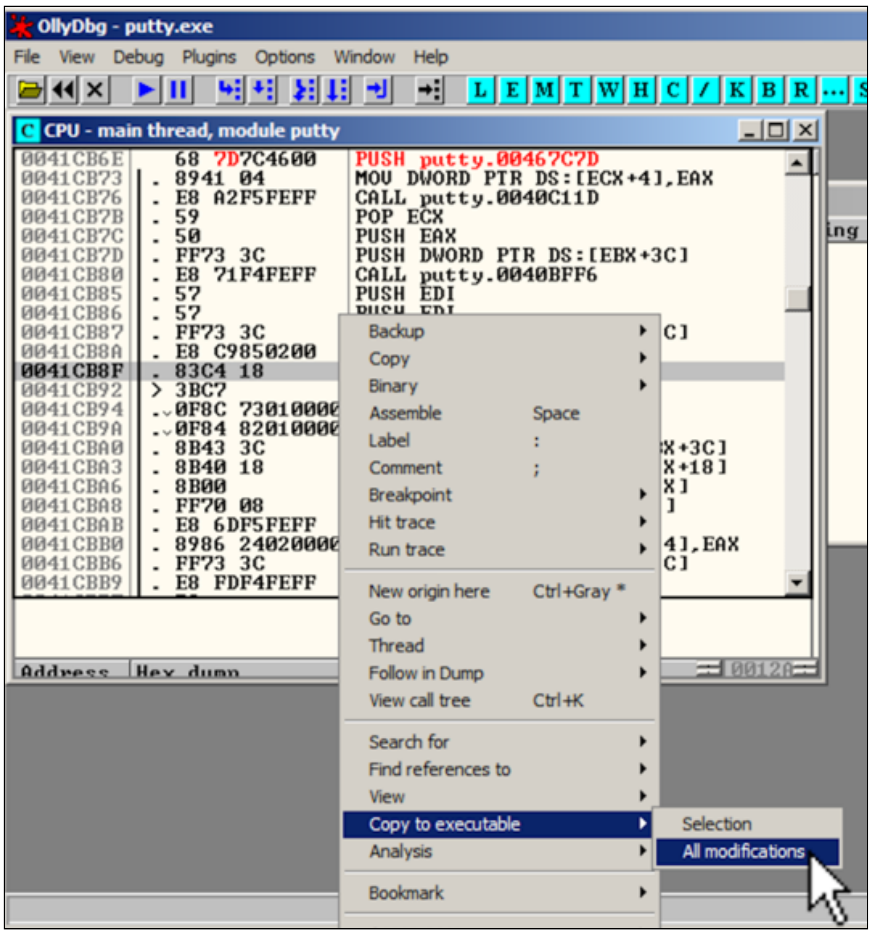


When I did it, an error box also popped up saying "Server unexpectedly closed network connection". If that happens, just close the error box.

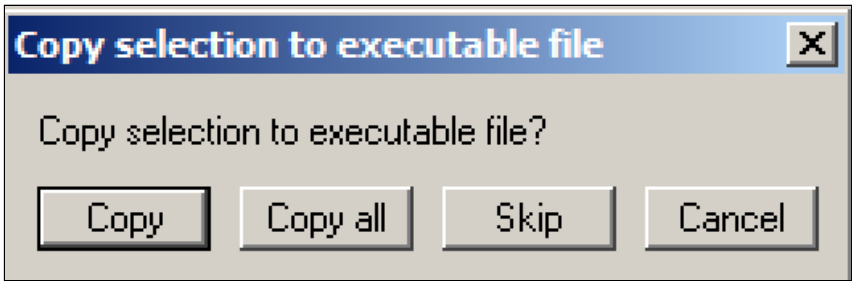
Saving the Modified .text Section

We have now changed an assembly language instruction; all executable code is in the **.text** section of the file.

In Ollydbg, in the top left pane of the CPU window, right-click, point to "**Copy to Executable**", and click "**All modifications**", as shown below.

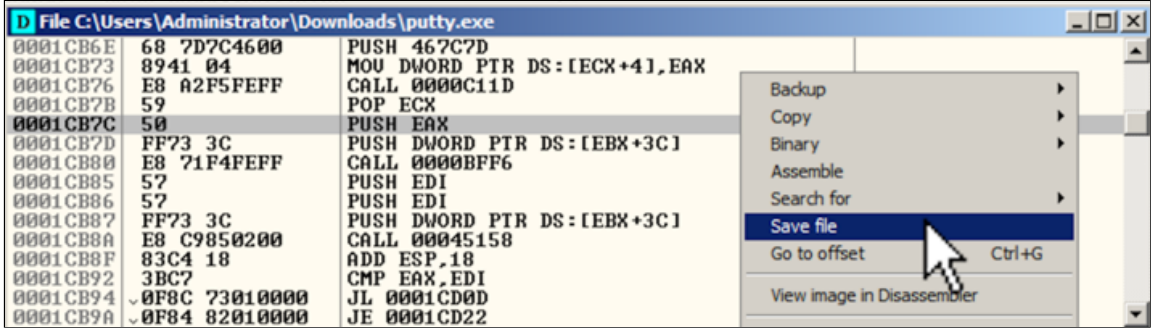


A "Copy selection to executable file" box pops up, as shown below. Click the "**Copy all**" button.



A new window pops up, with a title ending in "putty.exe", as shown below.

Right-click in the new window and click "**Save file**".



Save the file as **puttymod.exe**.

Running the Modified EXE

Close Ollydbg.

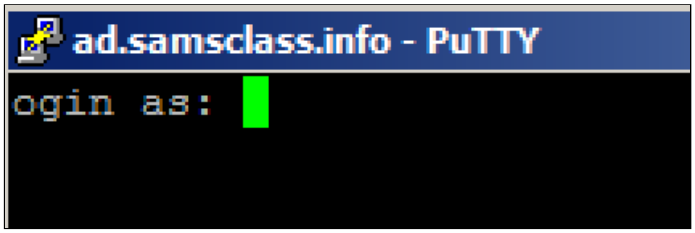
Double-click **puttymod.exe**.

In the "Host Name (or IP address)" box, type

```
ad.samsclass.info
```

At the bottom, click the **Open** button.

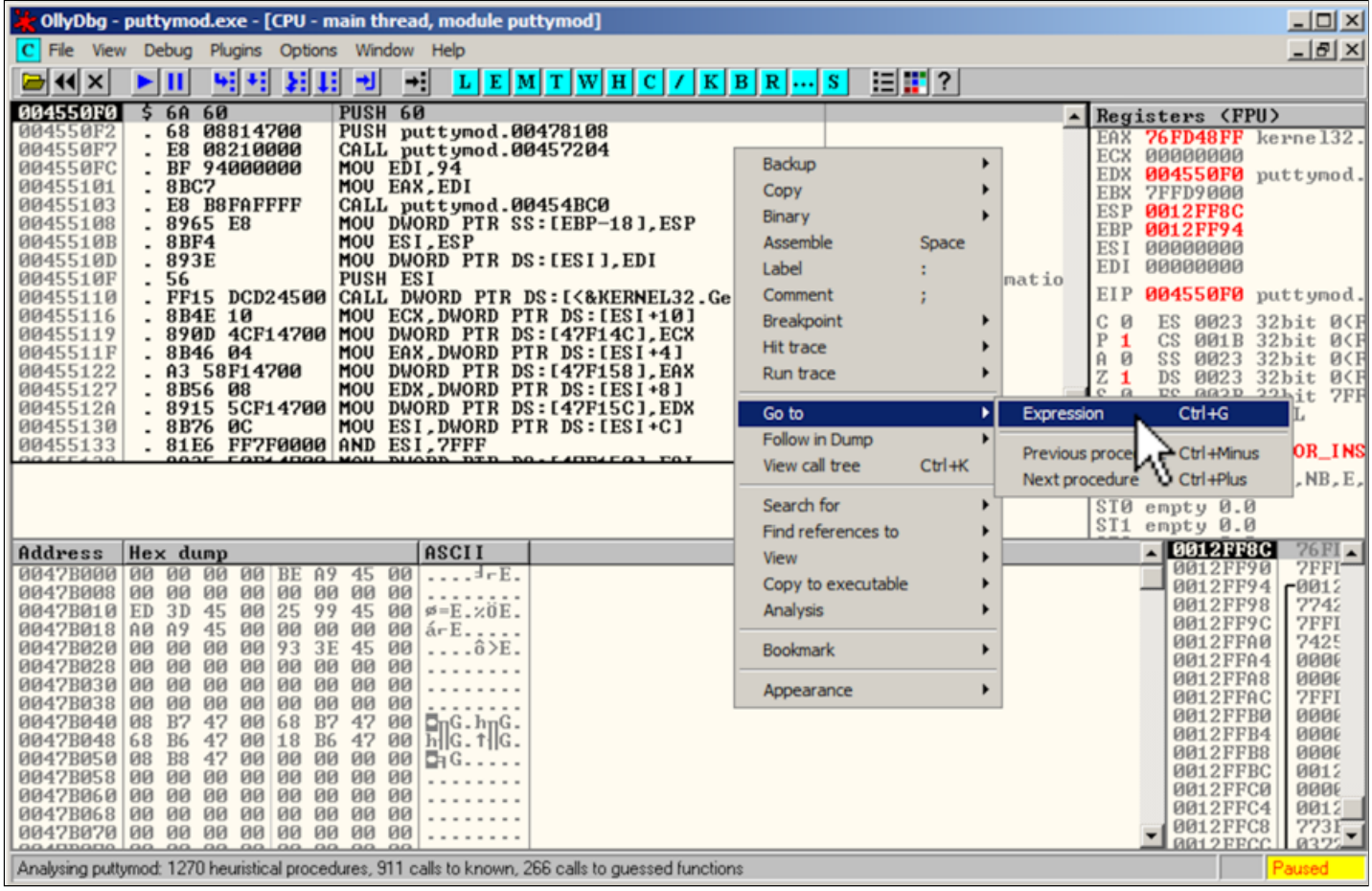
A black box opens, and shows a "**ogin as:**" prompt, as shown below.



Modifying the puttymod File

Open Ollydbg and load the **puttymod.exe** file.

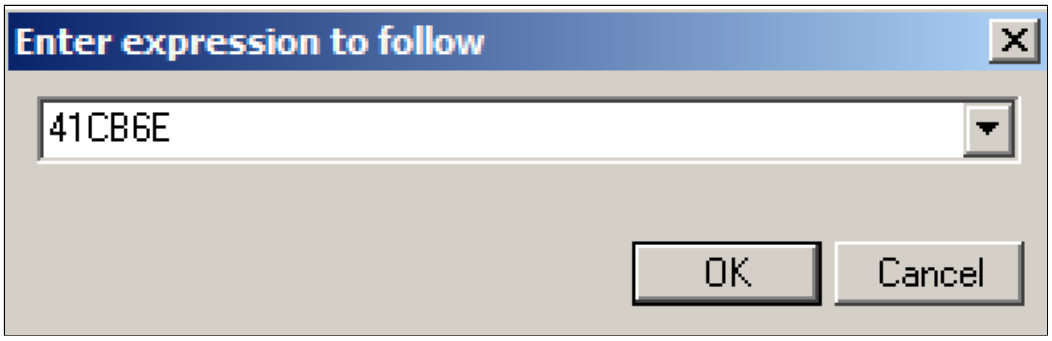
In the top left pane of the CPU window, right-click, point to "**Go to**", and click **Expression**, as shown below.



In the "Enter expression to follow" box, enter

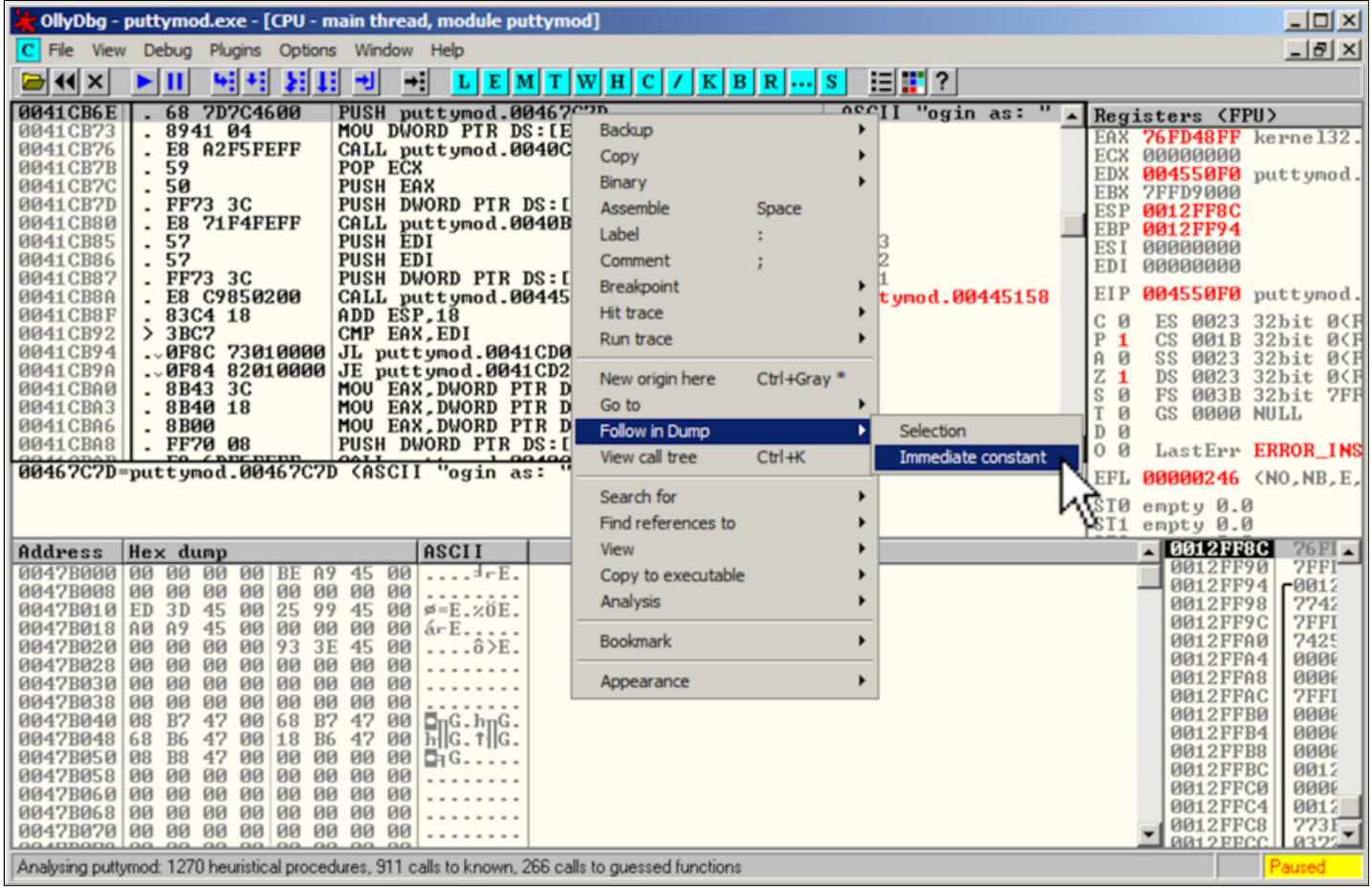
41CB6E

as shown below. Click **OK**.



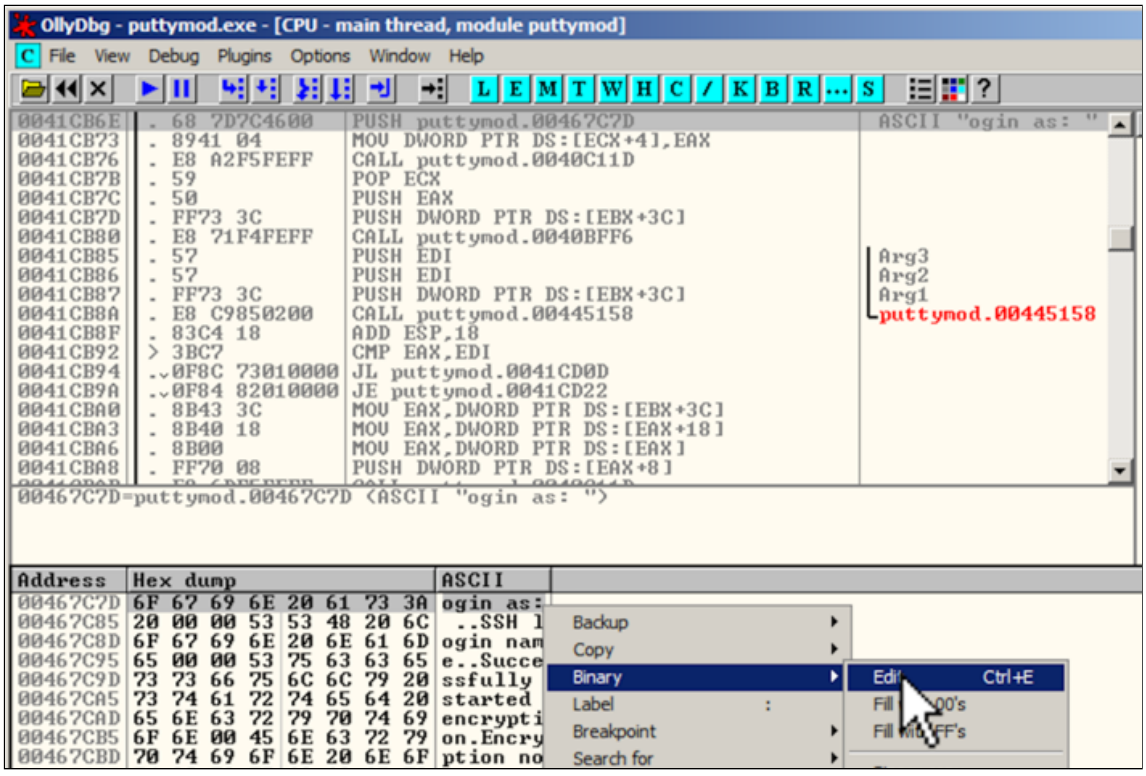
Changing the Login Message

In the top left pane of the CPU window, right-click **00467C7D**, as shown below. Point to "**Follow in Dump**" and click "**Immediate constant**".

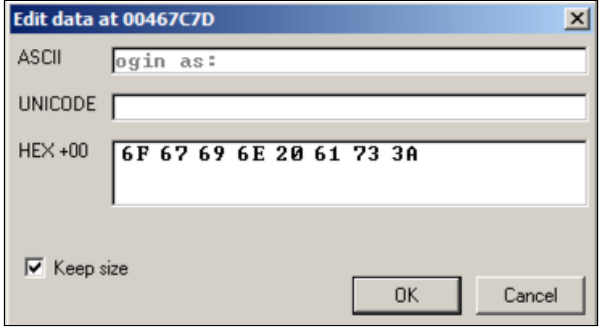


The Hex Dump pane, in the lower left, shows the text "**ogin as:** ", as shown below.

In the Hex Dump pane, highlight "**ogin as:**", as shown below. Right-click the highlighted text. Point to **Binary**. Click **Edit**.

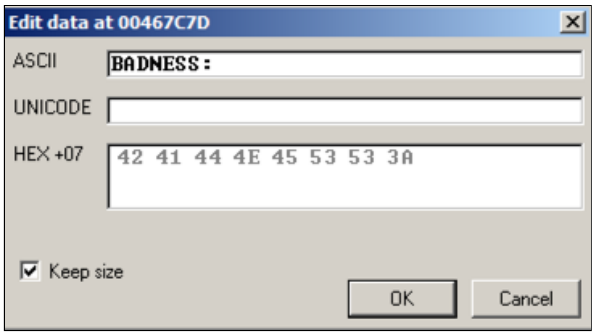


An "Edit data at 00467C7D" box opens, as shown below.

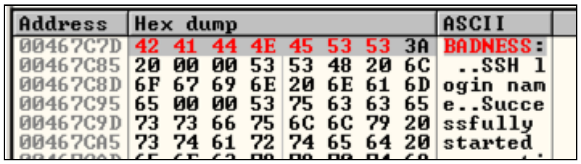


Click in the ASCII field, press Backspace to move back to the start, and overwrite the message with this text, as shown below:

BADNESS

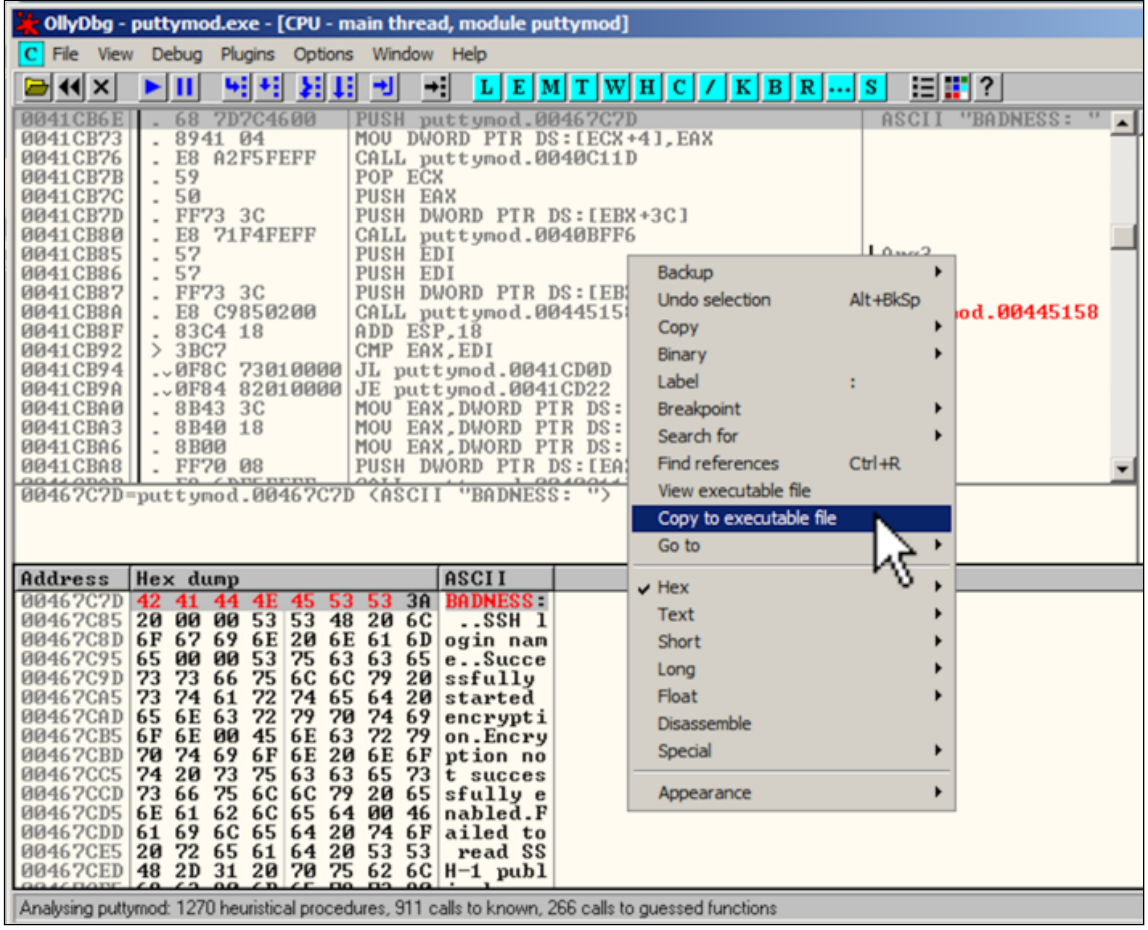


Click **OK**. The modified text appears in red letters in the Dump, as shown below.

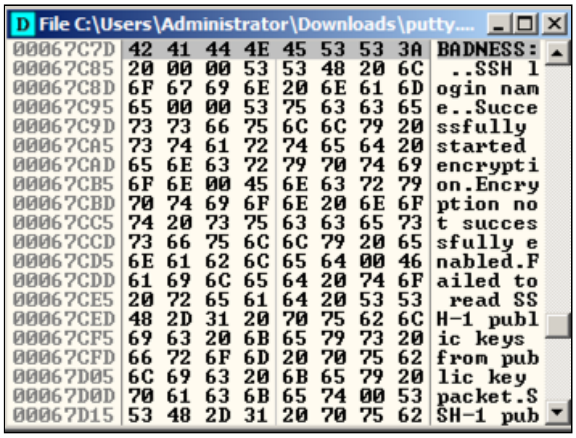


Saving the Modified ASCII Text

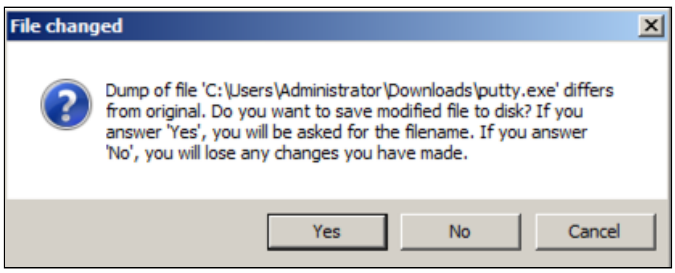
In Ollydbg, in the lower left "Dump" pane of the CPU window, right-click and click "**Copy to executable file**", as shown below.



A box with a long title ending in "puttymod.exe" appears showing the modified file, as shown below. Click the **X** in the top right of that box to close it.



A "File changed" box appears, as shown below. Click **Yes**.



Save the file as "puttymod2.exe".

Running the Modified EXE

Close Ollydbg.

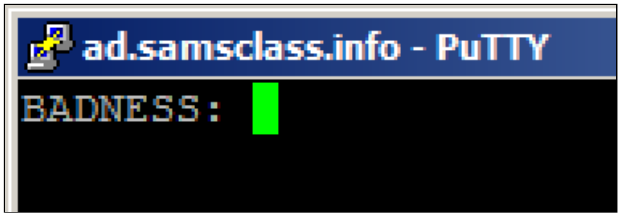
Double-click **puttymod2.exe**.

In the "Host Name (or IP address)" box, type

`ad.samsclass.info`

At the bottom, click the **Open** button.

A black box opens, and shows a "**BADNESS:**" prompt, as shown below.



PMA 401.1: Calculating the Hash (20 pts)

Calculate the CRC32 hash of **puttymod2.exe**

The flag is that hash like this: **`07b01710`**

Patching More EXEs

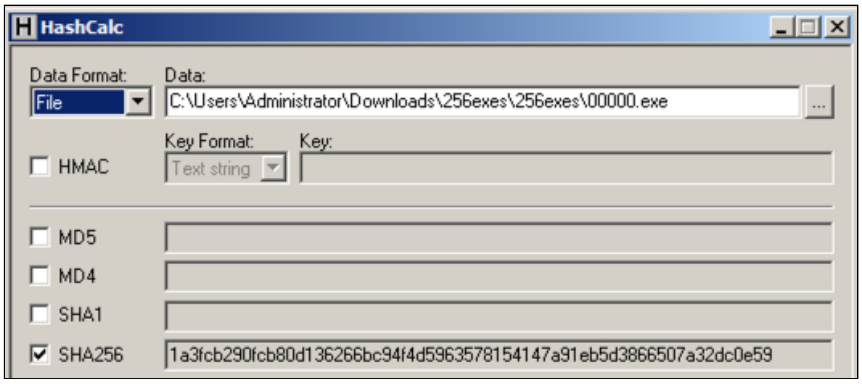
Getting the Files

You need several files to examine. Download them with these links:

- **00000.exe**
- **3EXEs.zip**
- **easy.zip**
- **256exes.zip**

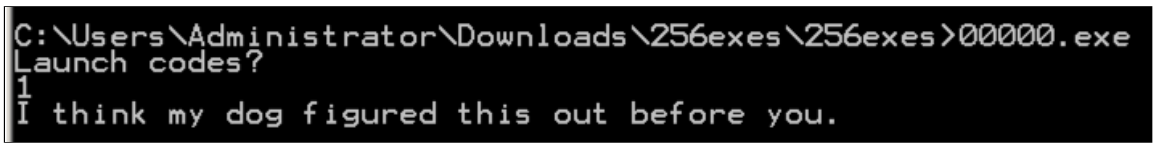
Checking the Hash

Calculate the SHA256 hash of the **00000.exe** file. It should match the value shown below.



Running 00000.exe

Run **00000.exe** in a Command Prompt. It asks for a "Launch code" and if you guess wrong, it insults you, as shown below.



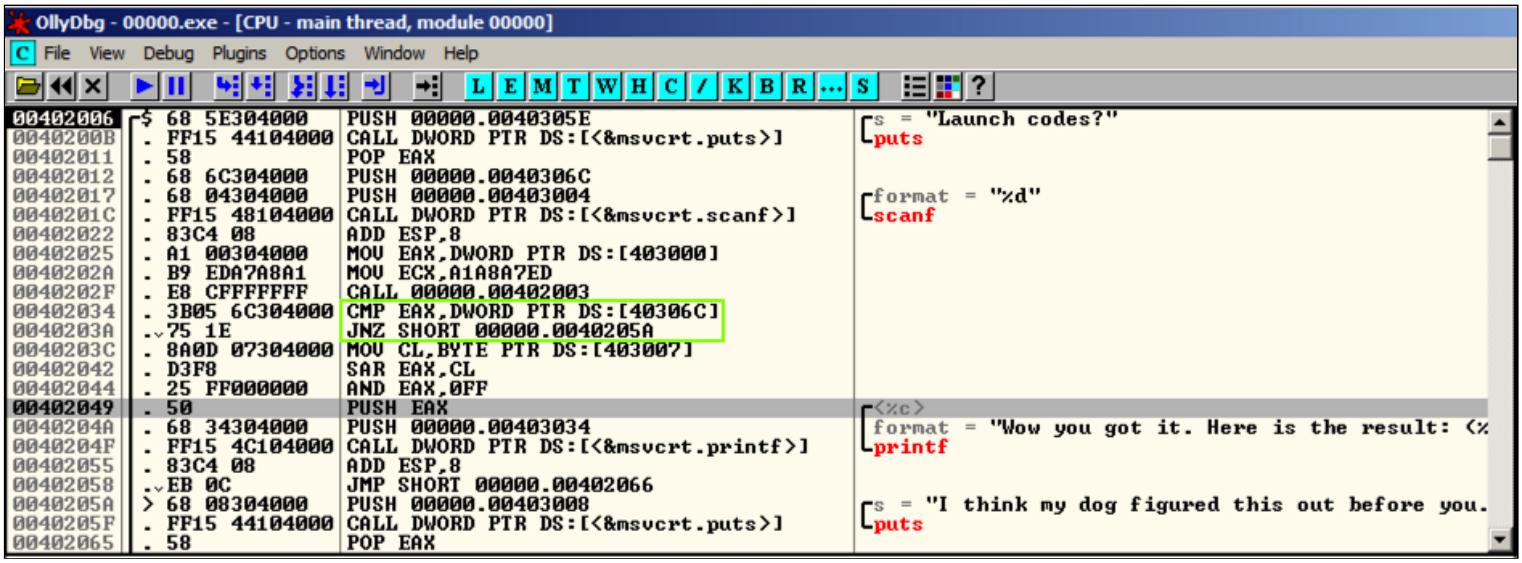
Examining the EXE with Ollydbg

Open the file in OllyDbg, as shown below.

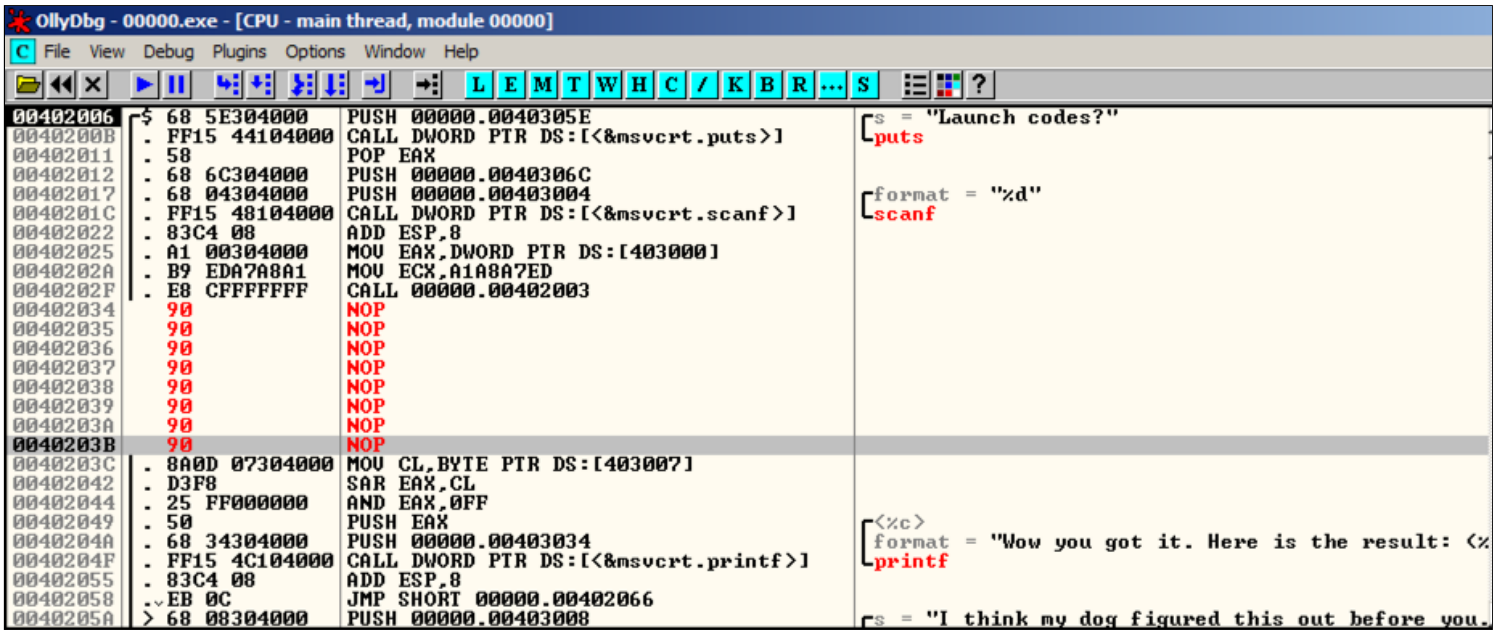
If you are using a 64-bit Windows system, click **Debug**, **Run** once to get to the start of the 32-bit code, as shown below.

In the assembly code pane, look at the rightmost column, and you can easily see what the program does; it prints out "Launch codes?", reads in a decimal number (%d), and then chooses to print either a winning message with a result, or an insult.

The choice is performed by two instructions: CMP (Compare) and JNZ (Jump if Not Zero), outlined in green in the image below.

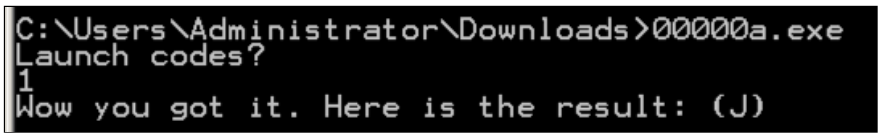


Replace those instructions with NOP (No Operation), as shown below. Save the patched file.



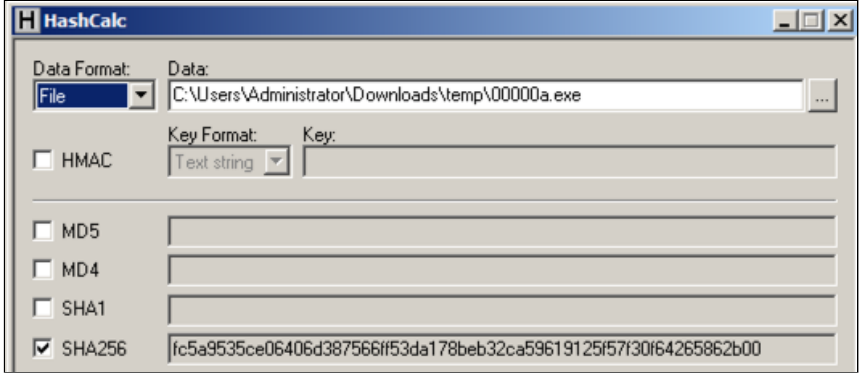
Running the Patched File

The patched file will accept any Launch code, as shown below.



Checking the Hash

Calculate the SHA256 hash of the patched file. It should match the value shown below.



PMA 401.2: CRC32 of Patched File (10 pts)

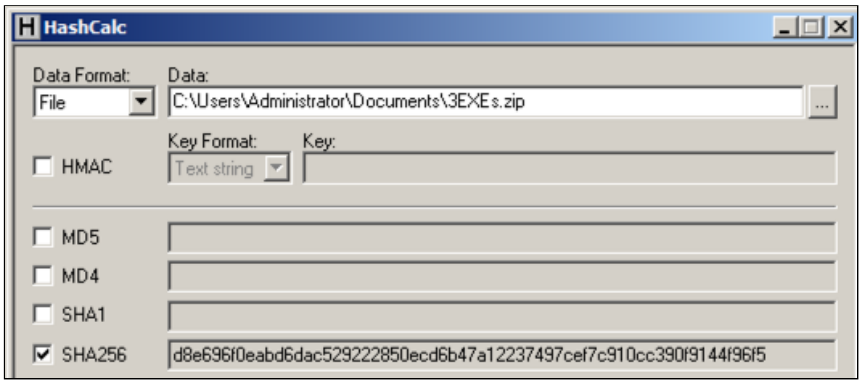
Calculate the CRC32 hash of the patched file.

The flag is that hash like this: **`07b01710`**

Patching Three EXEs

Checking the Hash

Calculate the SHA256 hash of the **3EXEs.zip** file. It should match the value shown below.



Patch the Files

Unzip the archive. Patch all 3 files so they will accept any input.

Gather the Results

Run the three patched files. Each one returns a single character as a result. Keep the files in alphabetical order, by filename, like this:

- File **00000.exe** Result **C**
- File **0000a.exe** Result **A**
- File **000a1.exe** Result **T**

If those were the results, the answer would be **CAT**

The actual results are different, of course.

PMA 401.3: Three Characters

The flag is the results, three characters like this: **`CAT`**

Patching 19 EXEs

Getting the EXEs

Unzip the **easy.zip** file. There are 19 EXEs in it.

Goal

Patch all 19 files, run them, and combine the Results to get a 19-character flag.

Hints

There are hints here.

PMA 401.4: Nineteen Characters

The flag is the results, 19 Characters like this: **`Impenetrable!Cyber!`**

Patching 256 EXEs

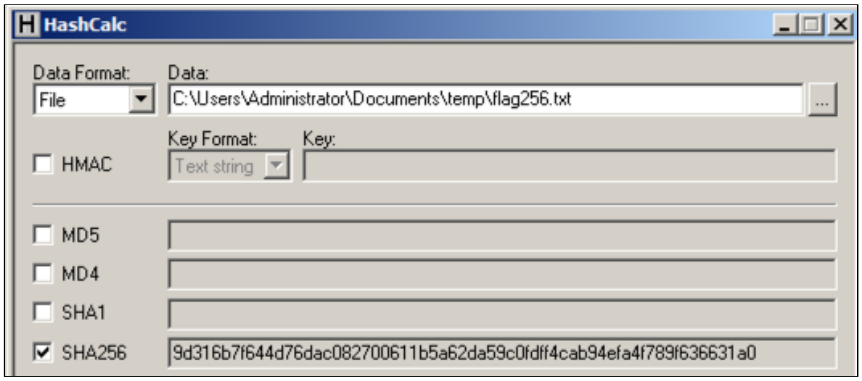
Getting the EXEs

Unzip the **256exes.zip** file. There are 256 EXEs in it.

Goal: Gather the Results

Patch all 256 files and run them. Each file will give you one "Result" character. Gather all those characters into a file 256 bytes long.

Calculate the SHA256 hash of that file. It should match the value shown below.



Calculate the CRC32 of that file to win.

PMA 401.5: CRC32

The flag is the CRC32 hash of the 256-byte file.

Credit

This is based on the 67k Challenge from EasyCTF 2017.

Sources

Backdooring PE Files - Part 1
Art of Anti Detection 2 â€“ PE Backdoor Manufacturing
https://github.com/EgeBalci/Cminer
https://en.wikipedia.org/wiki/Code_cave
http://stackoverflow.com/questions/787100/what-is-a-code-cave-and-is-there-any-legitimate-use-for-one
The Beginners Guide to Codecaves
Reversing with Ollydbg debugger
Ollydbg 'Copy all modifications to executable' doesn't copy all modifications

---

Revised 5-21-19 for WCIL
Download links moved to correct positions 9-8-2020

References to Win 2008 removed 2-23-21