Using the Bochs debugger plugin in Linux

Introduction

This guide illustrates how to configure the Bochs debugger plugin under Linux/MacOS.

Downloading and compiling Bochs

Please download the Bochs source code tarball and extract it.

```
tar zxf bochs-2.5.1.tar.gz
```

Run the 'configure' script (it is possible to pass other switches) and make sure that the switches marked in bold are present:

```
./configure --enable-sb16 --enable-ne2000 --enable-all-optimizations \
--enable-cpu-level=6 --enable-x86-64 --enable-pci \
 --enable-clgd54xx --enable-usb --enable-usb-ohci \
 --enable-plugins  --enable-show-ips --with-all-libs \
 --enable-debugger --disable-readline
```

Note: under MacOS Lion 10.7.3 use the following switches:

```
./configure --enable-cpu-level=6 --with-nogui --enable-debugger --enable-
disasm --enable-x86-debugger --enable-x86-64 --disable-readline --enable-all-
optimizations --enable-sb16 --enable-ne2000 --enable-pci --enable-acpi
--enable-clgd54xx --enable-usb --enable-usb-ohci --enable-plugins --enable-
show-ips
```

For a complete installation guide please check:
http://bochs.sourceforge.net/doc/docbook/user/compiling.html.

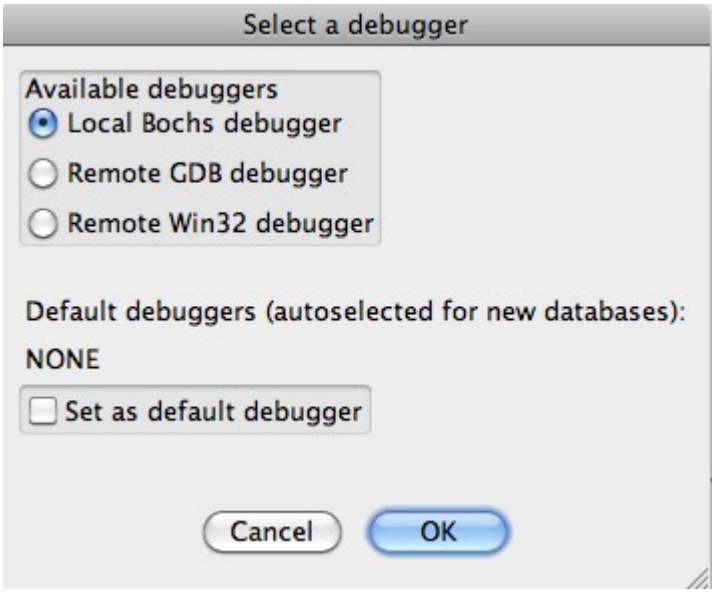
Now run "make" and "make install". Then type "whereis bochs" to get something like:

```
lallous@ubuntu:~/dev/bochs-2.5.1$ whereis bochs
bochs: /usr/local/bin/bochs /usr/local/lib/bochs
```

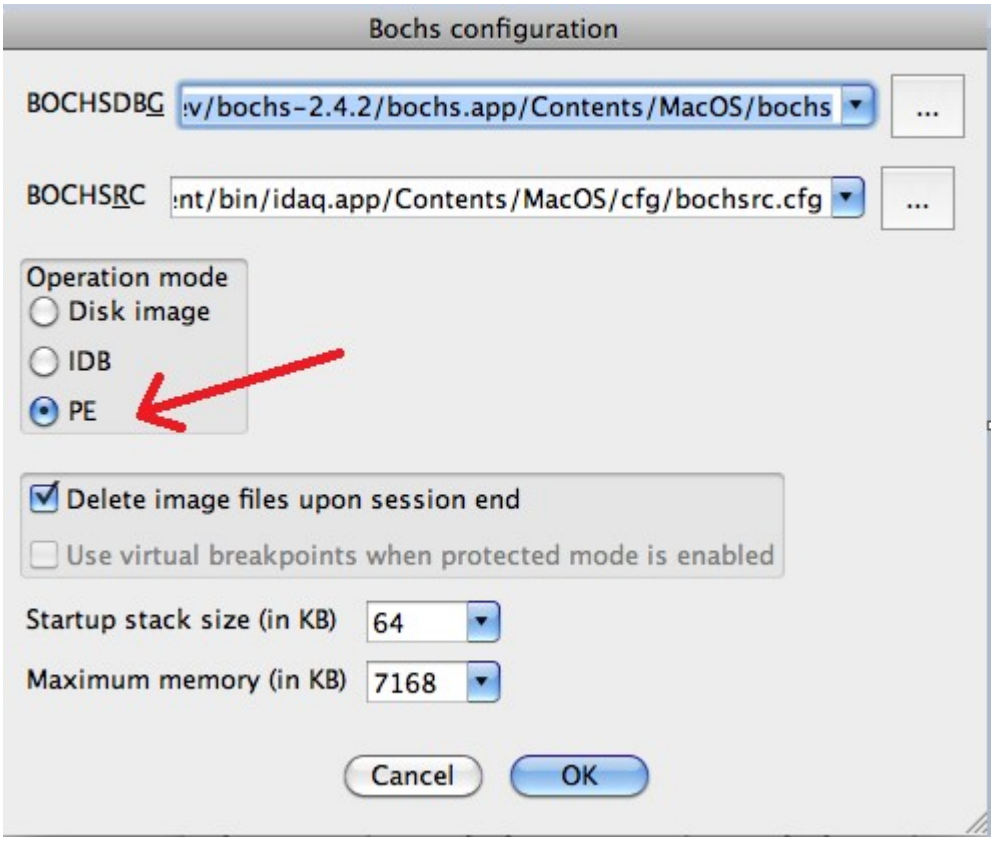
Configuring IDA and the Bochs debugger plugin

Opening a database and selecting the Bochs debugger

After installing Bochs, run IDA Pro and open a Windows PE file and select 'Debugger -> switch debugger' and select "Local Bochs Debugger":

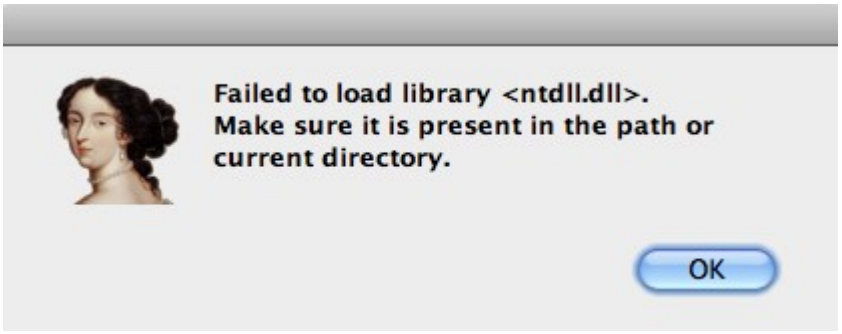


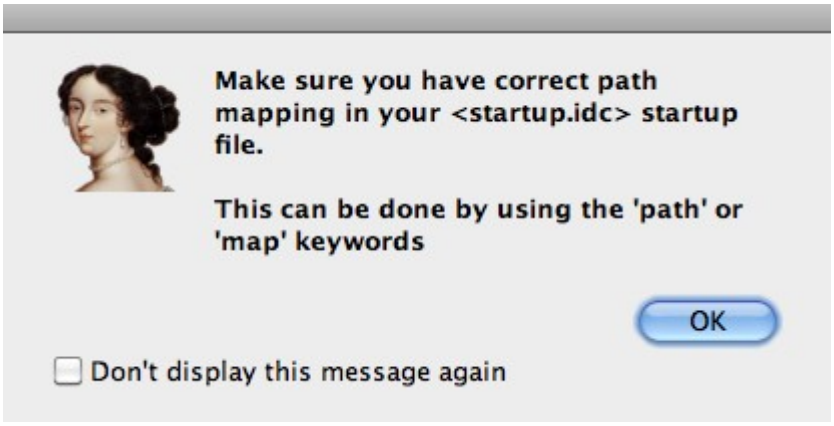
If a PE file was loaded, then the Bochs debugger plugin will operate in "PE mode":



In case the other two modes (IDB or Disk Image mode) are used then there is no need to specify any additional configurations options, otherwise please continue reading this guide.

Before launching the debugger with **F9**, the Bochs debugger plugin needs to know where to find the MS Windows DLLs and which environment variables to use. Attempting to run the debugger without configuring it may result in errors like this:





Here is a basic list of DLLs that are needed by most programs:

- advapi32.dll
- comctl32.dll
- comdlg32.dll
- gdi32.dll
- kernel32.dll
- msvcrt.dll
- mswsock.dll
- ntdll.dll
- ntoskrnl.exe
- shell32.dll
- shlwapi.dll
- urlmon.dll
- user32.dll
- wininet.dll
- ws2_32.dll
- wsock32.dll

Let us create a directory in $HOME/**bochs_windir**/ and place those DLLs there.

Specifying the Windows DLL path and environment variables using the startup file

The startup file is a script file found in idadir\plugins\bochs directory.

If IDC was the currently active language then **startup.idc** is used, otherwise startup**.ext** (where **ext** is the extension used by the currently selected extlang).

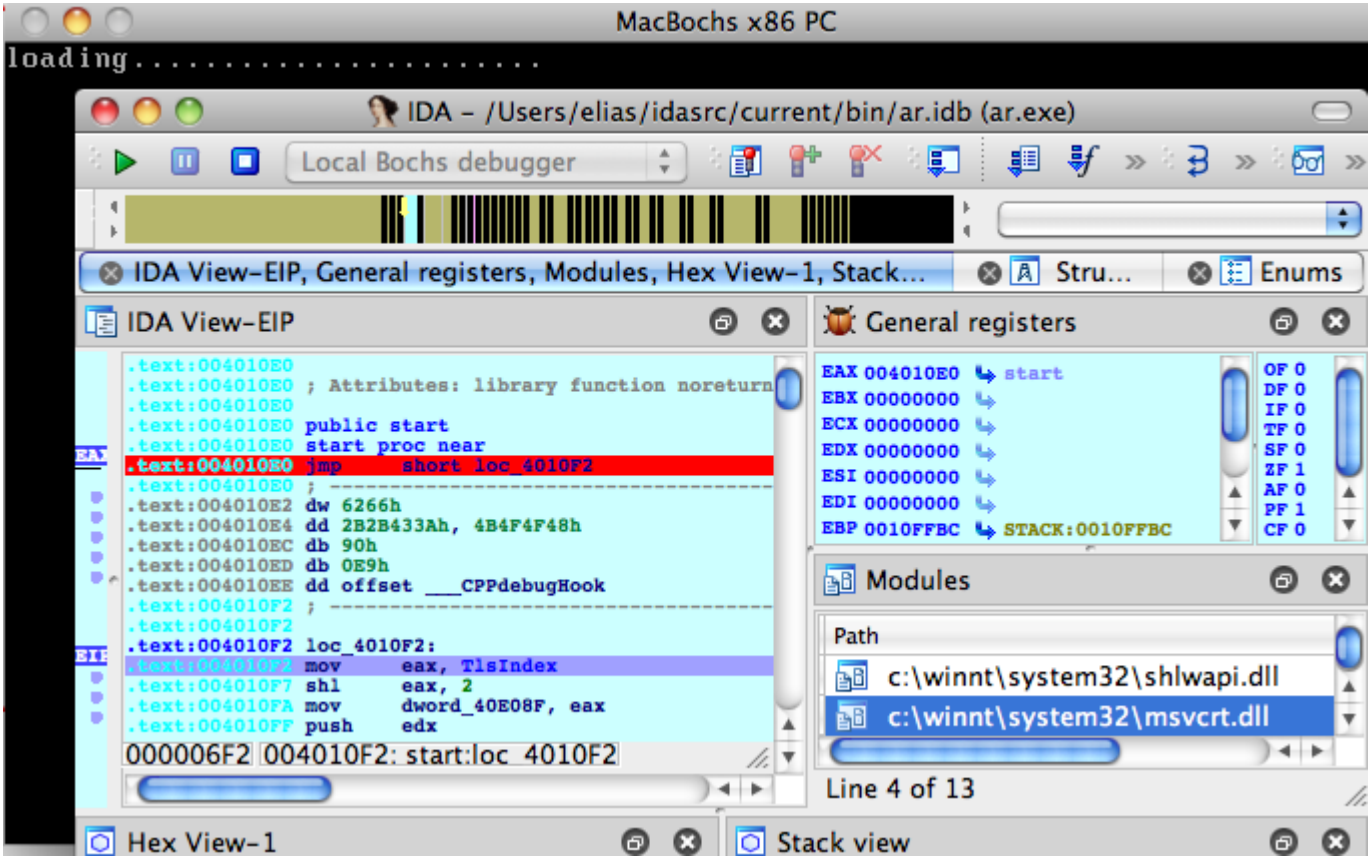
In this tutorial we will be working with IDC, so we will edit the **startup.idc** file. (Please note that changes to this file will affect all databases. For local changes (database specific configuration) take a copy of the startup script file and place it in the same directory as the database then modify it).

It is possible to specify a path map for a complete directory, for example:

```
/// path /home/lallous/bochs_windir/=c:\windows\system32
```

This line means that */home/lallous/bochs_windir/** will be visible to the debugged program as *c:\windows\system32** (for example */home/lallous/bochs_windir/ntdll.dll* will be visible as *c:\windows\system32\ntdll.dll*)

If all DLLs referenced by the program are in the **bochs_windir** directory, then running the process again should work:



(Bochs has already started and IDA switched to debugging mode.)

There are two things that should be configured. Press "." to switch to the output window (or use the Debugger / Modules list window to inspect the modules list):

```
E0000000: loaded /Users/elias/idasrc/current/bin/idaq.app/Contents/MacOS/plugins/bochs/bochsys.dll
77C60000: loaded c:\winnt\system32\advapi32.dll
7DAB0000: loaded c:\winnt\system32\gdi32.dll
7DD60000: loaded c:\winnt\system32\kernel32.dll
6FF50000: loaded c:\winnt\system32\msvcrt.dll
6C880000: loaded c:\winnt\system32\mswsock.dll
7DE70000: loaded c:\winnt\system32\ntdll.dll
73800000: loaded c:\winnt\system32\shell32.dll
6DE20000: loaded c:\winnt\system32\shlwapi.dll
72400000: loaded c:\winnt\system32\urlmon.dll
7DC50000: loaded c:\winnt\system32\user32.dll
71200000: loaded c:\winnt\system32\wininet.dll
400000: process /Users/elias/idasrc/current/bin/ar.exe has started (pid=3776)
```

1. The path to **bochsys.dll** is still not properly mapped. In our case, we need to add the following line to the startup file:

```
/// map
/Users/elias/idasrc/current/bin/idaq.app/Contents/MacOS/plugins/bochs/bochsys.dll=c:\windows\sy
stem32\bochsys.dll
```

   (As opposed to the **path** keyword that maps complete directories, the **map** keyword to map individual files)

   To hide the presence of bochsys.dll, simply map it to another name:

```
/// map
/Users/elias/idasrc/current/bin/idaq.app/Contents/MacOS/plugins/bochs/bochsys.dll=c:\windo
ws\system32\kvm.dll
```

2. The executable's path: we also need to add a **map** for the executable itself or a **path** entry for the whole folder:

```
/// path /Users/elias/idasrc/current/bin/=c:\malware
```

Now, after we run the program again we should get a more correct module list:

```
Bochs debugger has been terminated.
Debugger: process has exited (exit code 0)
E0000000: loaded c:\windows\system32\kvm.dll
77C60000: loaded c:\winnt\system32\advapi32.dll
7DAB0000: loaded c:\winnt\system32\gdi32.dll
7DD60000: loaded c:\winnt\system32\kernel32.dll
6FF50000: loaded c:\winnt\system32\msvcrt.dll
6C880000: loaded c:\winnt\system32\mswsock.dll
7DE70000: loaded c:\winnt\system32\ntdll.dll
73800000: loaded c:\winnt\system32\shell32.dll
6DE20000: loaded c:\winnt\system32\shlwapi.dll
72400000: loaded c:\winnt\system32\urlmon.dll
7DC50000: loaded c:\winnt\system32\user32.dll
71200000: loaded c:\winnt\system32\wininet.dll
400000: process c:\malware\ar.exe has started (pid=6892)
Bochs debugger has been initialized.
```

It is equally important to specify some environment variables. We will use the **env** keyword to define all the environment variables:

```
/// env PATH=c:\windows;c:\tools
/// env USERPROFILE=C:\Users\Guest
```

Specifying the Windows DLL path and environment variables using environment variables

An alternative way of configuring the DLLs path and environment variables is to use the **IDABXPATHMAP** and the **IDABXENVMAP** environment variables.

To specify the path map, export the following environment variable:

```
$ export IDABXPATHMAP=/home/lallous/bochs_windir/=c:/windows/system32;\
         /home/lallous/dev/idaadv/plugins/bochs/=c:/windows/system32;\
         /home/lallous/temp=c:/malware
```

(Please note that the forward slash (/) will be replaced with a backslash automatically by the plugin)

Similarly, specify the environment variables with the **IDABXENVMAP** environment variable:

```
$ export IDABXENVMAP="USERPROFILE=c:/Users/Guest++PATH=c:/windows;c:\tools"
```

(Please note that we used the ++ to separate between multiple variables)

In case you require to do specific changes (per database) to the startup file then please take a copy of it and place it in the same directory as the database. Refer to the help IDA Pro help file for more information.