Itanium I writeup

0xbadc0de

August 12, 2018

Abstract

This document refers to the write-up of the Itanium I challenge of the OpenCTF contest. Initially it was classified by the organisers as a pwning challenge, but later on its classification changed to reversing, crypto. In our point of view it is more related to a reversing challenge. This binary is in fact a VM interpreter of an extension of the brainfuck language.

1 Overview of the binary

The binary contains several options. All of them boil down to an execve call that reflects **itanium <file>**, where **<file>** is actually a path to one of the **.enc** files contained within the application directory. When executed with one argument that is a path to a **.enc** file it will decrypt it, allocate VM resources and start the VM execution.

1.1 Overview on the BF-VM

In this section I will only talk about the syntax extension done to the BF-VM. We assume that *ip* is the instruction pointer of the BF-VM. These extensions were:

1. ? - will read from the current *ip* a decryption key, lets call it x.
2. # - will read an upper bound for the decryption to occur, lets call it y.
3. * - will perform XOR decryption using x and from *ip* to $ip + y$.
4. 0x5e - Irrelevant for this challenge.
5. % - Irrelevant for this challenge.
6. ! - Chainloads another program from the socket.

1.2 Finding the flag

The challenge is about boarding the boat. Which seems to be a metaphore to chainloading your own VM file. There is only one **.enc** file that can do it, which is **dock.enc**. So You will find that the BF-VM for this challenge will calculate in memory the flag for the challenge. All you really need to do is to inspect the memory allocated by the DATA segment of the BF-VM which should be a heap pointer stored at VA: $0x6D3DC0$ plus the size of the VM code. Therefore you can then see something like:

0x0007F7906C54022 |"OpenCTF{You_got_the_encoding_keep_g01ng}"