PhenoProtect new-version Cracking (Zanzarah - German version)

GAME: Zanzarah – Das verborgene Portal (German version) [https://www.mobygames.com/game/7120/zanzarah-the-hidden-portal]
Protection: phenoProtect (new version)
Author: Luca D'Amico - https://www.lucadamico.dev - V1.0 - 27-Sep-2024

DISCLAIMER:

You will need:
- Windows 10/11 VM
- isdcc 1.22 [https://archive.org/download/InstallShieldTools/isdcc122.zip]
- isdcc 2.10 with mkcrc.exe
[https://ftp.area51.dev/pub/os2/hobbes.nmsu.edu/download/pub/windows/util/archiver/isDcc_2-10.zip]
- HxD [https://mh-nexus.de/en/downloads.php?product=HxD20]
- UniExtract v2 [https://github.com/Bioruebe/UniExtract2]
- Original game disc

Before you start:
I assume you've already read my first paper on phenoProtect (Carrera Grand Prix), so many details about this DRM are not covered here. If you haven't, I suggest you read it first and come back here right after. Once again, two different methods are presented to bypass this new version of phenoProtect.

Warning (for Method A): When I started removing phenoProtect from the installation script, I wasn't aware of the newer isdcc 2.10 tool, so I used version 1.22 (1998). It was much more difficult to match the InstallShield script's opcodes using this older version. Please, for the sake of your sanity, use version 2.10 instead: you will get a recompilable script (not relevant in this specific case) and a nice list of the decoded opcodes for every line!

phenoProtect trigger: You will get an error at 50% of installation when using a disc image.

METHOD A:
It is a bit more difficult to patch this version compared to the one found in Carrera Grand Prix due to the fact that the script is in .ins format. I wasn't able to find a tool like SID, so I had to use isdcc, which decompiled the script. Using isdcc version 1.22, you will not be able to compile this script back to its binary form using InstallShield 5 (many errors during compilation). Therefore, I had to hex-edit the file after finding the correct opcodes.
This was challenging because I had no prior knowledge of InstallShield scripts from a binary perspective. Fortunately, I found an excellent paper written by a very skilled cracker:
https://www.darkridge.com/~jpr5/mirror/fravia.org/natz51.htm
The interesting part is in the "second approach" chapter of the document.
Now, we know that an "if" compare instruction starts with the following bytes:

28, 01, 32

In the decompiled script, if you look closely, you will find a string called **"COPYPROTECTION"**.

Here is the relevant decompiled code:

```
begin
        Enable(4);
        StrLoadString("", "COPYPROTECTION", lString0);
        SetStatusWindow(0, lString0);
        Delay(2);
        Disable(54);
        StatusUpdate(1, 100);
        function109();
        lNumber2 = LAST_RESULT;
        lNumber2 = lNumber2 = 0;
        if (lNumber2 = 0) then
            goto label775;
        endif;
        StrLoadString("", "COPYPROTECTION_FAILED", lString0);
        MessageBox(lString0, -65533);
        number34 = 1;
        return(-1);

label775: //Ref: 00DF73
        StrLoadString("", "COPY_PROGRAM_FILES", lString0);
        SetStatusWindow(0, lString0);
        lNumber1 = 1;
        ComponentMoveData(MEDIA, lNumber1, 0);
        lNumber0 = LAST_RESULT;
        function112(lNumber0);
        Disable(4);
        return(lNumber0);
        return;
end;
```

Pay attention to the "**if (lNumber2 = 0)**" condition: if that statement is not satisfied, we will get the **COPYPROTECTION_FAILED** error.

We need to invert this statement to bypass the check, so we need to patch the previous line of code like this:

lNumber2 = lNumber2 != 0;

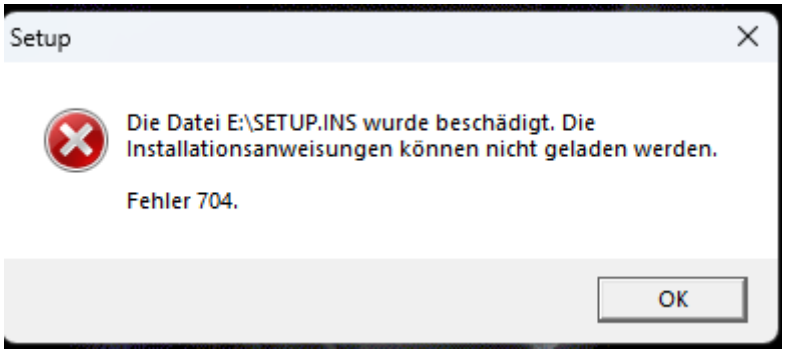
Using a hex editor (**HxD**) and searching for **"COPYPROTECTION_FAILED"**, we will arrive here:

```
0000DEF0  2F 01 B7 00 41 00 00 00 00 B8 00 00 00 10 00 B6  /.·.A.....¸.....¶
0000DF00  00 10 00 01 00 FF FF 00 00 03 00 00 00 01 00 41  .....ÿÿ........A
0000DF10  04 00 00 00 12 01 61 00 00 61 0E 00 43 4F 50 59  ......a..a..COPY
0000DF20  50 52 4F 54 45 43 54 49 4F 4E 52 9B FF 08 00 41  PROTECTIONR›ÿ..A
0000DF30  00 00 00 00 62 9B FF 0A 00 41 02 00 00 00 02 00  ....b›ÿ..A......
0000DF40  41 36 00 00 00 64 00 41 01 00 00 00 41 64 00 00  A6...d.A....Ad..
0000DF50  00 B5 00 80 6D 00 70 25 03 21 00 32 99 FF 42 00  .µ.€m.p%.!.2™ÿB.
0000DF60  00 28 01 32 99 FF 42 99 FF 41 05 00 00 00 41 00  .(.2™ÿB™ÿA....A.
0000DF70  00 00 00 22 00 70 07 03 95 42 99 FF 41 00 00 00  ...".p..•B™ÿA...
0000DF80  00 12 01 61 00 00 61 15 00 43 4F 50 59 50 52 4F  ...a..a..COPYPRO
0000DF90  54 45 43 54 49 4F 4E 5F 46 41 49 4C 45 44 52 9B  TECTION_FAILEDR›
0000DFA0  FF 2A 00 62 9B FF 41 03 00 FF FF 21 00 32 28 00  ÿ*.b›ÿA..ÿÿ!.2(.
0000DFB0  41 01 00 00 00 2F 01 B7 00 41 FF FF FF FF 00 00  A..../.·.Aÿÿÿÿ..
```

If we look closely, we will spot the **28, 01, 32** sequence: it marks the start of our "if" statement!

```
0000DF50  00 B5 00 80 6D 00 70 25 03 21 00 32 99 FF 42 00  .µ.€m.p%.!.2™ÿB.
0000DF60  00 28 01 32 99 FF 42 99 FF 41 05 00 00 00 41 00  .(.2™ÿB™ÿA....A.
0000DF70  00 00 00 22 00 70 07 03 95 42 99 FF 41 00 00 00  ...".p..•B™ÿA...
```

At address DF6A, there is a 0x5. If we patch this value with 0x6, we will invert the condition.
After applying this patch, if we decompile again, we will get:
lNumber2 = lNumber2 != 0;
Now we can overwrite the original setup.ins with our patched version and build a new ISO.
Let's install the game and... error:



After translating this error in Google Translate, we'll see that the installer is not happy because our patched setup.ins is "corrupted".
Since we patched this file, we probably broke some CRC.
I wasted a lot of time figuring out this issue. After hours, I realized that .ins files have a CRC value in their header.
At this point, after reading a Chinese forum, I discovered that a newer version of isdcc (version 2.10) not only decompiles .ins scripts producing a result accurate enough to be recompilable, but also provides an executable named mkcrc.exe that automatically calculates and updates the new CRC value in patched scripts. Moreover, this version provides a 1:1 opcode-to-decompiled code output, so you can avoid wasting time figuring out the exact opcode to patch.

So, let's run mkcrc.exe on our patched script and rebuild the new ISO.
This time everything works as expected, and we can install the game.

METHOD B:
Nothing new here. You can still extract .cab files using UniExtract to obtain the game's files. The main game directory is in data1.cab. Just remember to create a "save" directory in the root directory (where Data, Resource and System directories were extracted), otherwise, you will not be able to save in-game.

Credits:
As in my previous paper: I'd like to thank Codecult/Phenomedia AG for creating this unique protection scheme.
A big thank you also goes to the authors of the tools used to perform this analysis.

Conclusion:
This second iteration of phenoProtect wasn't particularly challenging, but it's still fun to patch! :)
Thank you for reading!
Luca