

# MDK v1.1

by BuXXe

## Installation

- Place the four EXE files in the main MDK folder.
- Use the correct mdkfont.fti for your language and place it with the LOAD\_0.LBB file into the MISC folder.
- Finally, it is recommended to set the hduse value in the MDK.CFG file to 6.

## Features

- Extended sound buffers allow higher quality sound files
- Start any level in the TRAVERSE folder directly from the main menu

## Limitations / Known Bugs

- level folder name is limited to 8 characters which means only 3 digits after "LEVEL"
- level numbers are limited to one byte (255) due to the BYTE array level lookup
- level numbers up to 8 (inclusive) are reserved
  - ⇒ custom levels can only have LEVEL[9-255] as folder names
- after a custom level is completed, the game continues with the stream sequence of LEVEL8 and starts LEVEL8 afterwards.
- Saving a game in a custom level will produce a .sav file which cannot be loaded yet. This is due to the fact that the save file stores the index of the level in it instead of the level number. Custom levels are always put at the end of the level lookup list which will give all custom level save game files the same level index. A future solution would be to rework the save game file format to use the level numbers instead of indices.

## Patch Base

The GOG release will be declared as version 1.0. This patch is based on the MDK95.exe, MDKD3D.exe and MDK3DFX.exe of that release. The additional Mdka3d3d.exe file is taken from [https://www.infania.net/misc/gona/3D/files/MDK/mdk\\_patches.html](https://www.infania.net/misc/gona/3D/files/MDK/mdk_patches.html) [MDK\_A3D\_Direct3D\_patch\_(Force-Feedback\_patch\_compatible).zip (1998-Apr-17)]. The mdkfont.fti files for each Language are also taken from the GOG release.

The relocation tables were fixed manually to support the patch changes.

## Contact or support me



buxxe



**Discord**

[Link](#)



**PayPal**

[diebuxxe](#)

## MDK.CFG

The configuration file has an entry which is used to determine if the game is installed "fully". If not, the game will copy necessary files for the current level into the SCRATCH folder. This results in a longer loading time. Generally, users should have the full release installed and therefor be able to use the "full" install setting. It seems that the GOG release does not have this value set correctly. The correct value in the file needs to be: "hduze = 6"

## MISC/mdkfont.fti

The mdkfont.fti files for each language are extended by the entries "OPT5", "LN", "LT". Furthermore, the OPT entries are rearranged to form "Continue", "New Game", "Saved Games", "Level Select", "Options", "Quit".

## MISC/LOAD\_0.LBB

A new loading screen file is added for custom levels. For the sake of simplicity, there is only one version of the loading image instead of localized ones for each language.



## MDK executable file

The patch is applied to all 4 exe files. This changelog will only document the changes for one exe file in order to outline the reason for each change. Each exe has a slightly different code but the general ideas are the same.

### Data Segment:

Original	Patched
00 00 00 07 00 00 00 06 00 00 00 03 00 00 00 04 00 00 00 08 00 00 00 05 00 00 00 02 00 00 00 01 {DWORD-array level lookup table}	07 06 03 04 08 05 02 01 00 53 45 5C 4C 45 56 45 4C 2A 00 54 52 41 56 45 52 00 4C 54 00 4C 4E 00 {BYTE-array level lookup table} {level select; find path part 1} "SE\LEVEL*" {level select; find path part 2} "TRAVER" {level select; screen title text identifier} "LT" {level select; no levels found identifier} "LN"

### Code Segment:

Extended main memory and sound buffers

Main memory allocation

Original	Patched
SHL ECX, 0A	SHL ECX, 0E
{0x27F6 << 0x0A = 10.475.520 bytes}	{0x27F6 << 0x0E = 167.608.320 bytes}

Sound buffer allocation / size check

Original	Patched
MOV EAX, 7D000	MOV EAX, 5F5E100
...	...
CMP EAX, 7D000	CMP EAX, 5F5E100
0x7D000 = 512.000 bytes	0x5F5E100 = 100.000.000 bytes

### Main menu rendering optimization and version number rendering

For reference, the original code from the MDK95.EXE is displayed below. In it, the main menu entries are loaded from the mdkfont.fti file onto the stack and in registers. The entries have the identifiers "OPT0" to "OPT4". Next, the text

entries are used to compute their rendering width in pixels. The maximum width is divided by 2 and later used to render the main menu items centered horizontally. The code then checks if the "Continue" entry needs to be rendered and splits the execution path depending on the result. While rendering the menu entries, the code checks which entry is currently selected. This property is used by the rendering function to create the scaling effect present in the main menu for selected items. This part of the code is pretty long and provides a lot of optimization potential. The loading of the text entries and the calculation of the max width does not use a loop. The same is true for the rendering of the menu entries which is even worse because the two different cases for "With Continue" and "Without Continue" are realized independently.

## Original code listing

Address	Hex dump	Command	Comments
0041E31B	> \B8 EC594900	MOV EAX,OFFSET 004959EC	; ASCII "OPT0"
0041E320	. E8 6B65FFFF	CALL 00414890	; [MDK95.00414890
0041E325	. 8945 D4	MOV DWORD PTR SS:[LOCAL.11],EAX	
0041E328	. E8 BB68FFFF	CALL 00414BE8	; [MDK95.00414BE8
0041E32D	. 8945 D8	MOV DWORD PTR SS:[LOCAL.10],EAX	
0041E330	. B8 F4594900	MOV EAX,OFFSET 004959F4	; ASCII "OPT1"
0041E335	. E8 5665FFFF	CALL 00414890	; [MDK95.00414890
0041E33A	. 8945 DC	MOV DWORD PTR SS:[LOCAL.9],EAX	
0041E33D	. E8 A668FFFF	CALL 00414BE8	; [MDK95.00414BE8
0041E342	. 89C1	MOV ECX,EAX	
0041E344	. B8 FC594900	MOV EAX,OFFSET 004959FC	; ASCII "OPT2"
0041E349	. E8 4265FFFF	CALL 00414890	; [MDK95.00414890
0041E34E	. 89C7	MOV EDI,EAX	
0041E350	. E8 9368FFFF	CALL 00414BE8	; [MDK95.00414BE8
0041E355	. 89C3	MOV EBX,EAX	
0041E357	. B8 045A4900	MOV EAX,OFFSET 00495A04	; ASCII "OPT3"
0041E35C	. E8 2F65FFFF	CALL 00414890	; [MDK95.00414890
0041E361	. 8945 E0	MOV DWORD PTR SS:[LOCAL.8],EAX	
0041E364	. E8 7F68FFFF	CALL 00414BE8	; [MDK95.00414BE8
0041E369	. 89C6	MOV ESI,EAX	
0041E36B	. B8 0C5A4900	MOV EAX,OFFSET 00495A0C	; ASCII "OPT4"
0041E370	. 89CA	MOV EDX,ECX	
0041E372	. E8 1965FFFF	CALL 00414890	; [MDK95.00414890
0041E377	. 8945 E4	MOV DWORD PTR SS:[LOCAL.7],EAX	
0041E37A	. E8 6968FFFF	CALL 00414BE8	; [MDK95.00414BE8
0041E37F	. 39CE	CMP ESI,ECX	
0041E381	. 7E 02	JLE SHORT 0041E385	
0041E383	. 89F2	MOV EDX,ESI	
0041E385	> 39D0	CMP EAX,EDX	
0041E387	. 7E 02	JLE SHORT 0041E38B	
0041E389	. 89C2	MOV EDX,EAX	
0041E38B	> 39D3	CMP EBX,EDX	
0041E38D	. 7E 02	JLE SHORT 0041E391	
0041E38F	. 89DA	MOV EDX,EBX	
0041E391	> 8B45 D8	MOV EAX,DWORD PTR SS:[LOCAL.10]	
0041E394	. 39C2	CMP EDX,EAX	
0041E396	. 7D 02	JGE SHORT 0041E39A	
0041E398	. 89C2	MOV EDX,EAX	
0041E39A	> 89D0	MOV EAX,EDX	
0041E39C	. C1FA 1F	SAR EDX,1F	
0041E39F	. 2BC2	SUB EAX,EDX	
0041E3A1	. D1F8	SAR EAX,1	
0041E3A3	. 8B15 98BC5400	MOV EDX,DWORD PTR DS:[54BC98]	
0041E3A9	. 89C6	MOV ESI,EAX	
0041E3AB	. 85D2	TEST EDX,EDX	
0041E3AD	. 0F84 B9000000	JZ 0041E46C	
0041E3B3	. 8B4D D4	MOV ECX,DWORD PTR SS:[LOCAL.11]	
0041E3B6	. 89C2	MOV EDX,EAX	
0041E3B8	. A1 78AA4900	MOV EAX,DWORD PTR DS:[49AA78]	
0041E3BD	. BB 1F000000	MOV EBX,1F	
0041E3C2	. 85C0	TEST EAX,EAX	
0041E3C4	. 0F85 83000000	JNZ 0041E44D	
0041E3CA	. B8 01000000	MOV EAX,1	
0041E3CF	> E8 64570000	CALL 00423B38	; [MDK95.00423B38
0041E3D4	. 8B4D DC	MOV ECX,DWORD PTR SS:[LOCAL.9]	
0041E3D7	. BB 43000000	MOV EBX,43	
0041E3DC	. A1 78AA4900	MOV EAX,DWORD PTR DS:[49AA78]	
0041E3E1	. 89F2	MOV EDX,ESI	
0041E3E3	. 83F8 01	CMP EAX,1	

0041E3E6	. 75 6C	JNE SHORT 0041E454	
0041E3E8	> E8 4B570000	CALL 00423B38	; [MDK95.00423B38
0041E3ED	. BB 67000000	MOV EBX,67	
0041E3F2	. 89F9	MOV ECX,EDI	
0041E3F4	. 8B3D 78AA4900	MOV EDI,DWORD PTR DS:[49AA78]	
0041E3FA	. 89F2	MOV EDX,ESI	
0041E3FC	. 83FF 02	CMP EDI,2	
0041E3FF	. 75 57	JNE SHORT 0041E458	
0041E401	. B8 01000000	MOV EAX,1	
0041E406	> E8 2D570000	CALL 00423B38	; [MDK95.00423B38
0041E40B	. 8B4D E0	MOV ECX,DWORD PTR SS:[LOCAL.8]	
0041E40E	. BB 8B000000	MOV EBX,8B	
0041E413	. A1 78AA4900	MOV EAX,DWORD PTR DS:[49AA78]	
0041E418	. 89F2	MOV EDX,ESI	
0041E41A	. 83F8 03	CMP EAX,3	
0041E41D	. 75 3D	JNE SHORT 0041E45C	
0041E41F	. B8 01000000	MOV EAX,1	
0041E424	> E8 0F570000	CALL 00423B38	; [MDK95.00423B38
0041E429	. 8B4D E4	MOV ECX,DWORD PTR SS:[LOCAL.7]	
0041E42C	. 89F2	MOV EDX,ESI	
0041E42E	. 8B35 78AA4900	MOV ESI,DWORD PTR DS:[49AA78]	
0041E434	. BB AF000000	MOV EBX,0AF	
0041E439	. 83FE 04	CMP ESI,4	
0041E43C	. 75 22	JNE SHORT 0041E460	
0041E43E	. B8 01000000	MOV EAX,1	
0041E443	. E8 F0560000	CALL 00423B38	; [MDK95.00423B38
0041E448	. ^ E9 10FEFFFF	JMP 0041E25D	
0041E44D	> 31C0	XOR EAX,EAX	
0041E44F	. ^ E9 7BFFFFFF	JMP 0041E3CF	
0041E454	> 31C0	XOR EAX,EAX	
0041E456	. ^ EB 90	JMP SHORT 0041E3E8	
0041E458	> 31C0	XOR EAX,EAX	
0041E45A	. ^ EB AA	JMP SHORT 0041E406	
0041E45C	> 31C0	XOR EAX,EAX	
0041E45E	. ^ EB C4	JMP SHORT 0041E424	
0041E460	> 31C0	XOR EAX,EAX	
0041E462	. E8 D1560000	CALL 00423B38	; [MDK95.00423B38
0041E467	. ^ E9 F1FDFFFF	JMP 0041E25D	
0041E46C	> 8B4D DC	MOV ECX,DWORD PTR SS:[LOCAL.9]	
0041E46F	. 89C2	MOV EDX,EAX	
0041E471	. A1 78AA4900	MOV EAX,DWORD PTR DS:[49AA78]	
0041E476	. BB 1F000000	MOV EBX,1F	
0041E47B	. 83F8 01	CMP EAX,1	
0041E47E	. 75 65	JNE SHORT 0041E4E5	
0041E480	> E8 B3560000	CALL 00423B38	; [MDK95.00423B38
0041E485	. BB 43000000	MOV EBX,43	
0041E48A	. 89F9	MOV ECX,EDI	
0041E48C	. 8B3D 78AA4900	MOV EDI,DWORD PTR DS:[49AA78]	
0041E492	. 89F2	MOV EDX,ESI	
0041E494	. 83FF 02	CMP EDI,2	
0041E497	. 75 50	JNE SHORT 0041E4E9	
0041E499	. B8 01000000	MOV EAX,1	
0041E49E	> E8 95560000	CALL 00423B38	; [MDK95.00423B38
0041E4A3	. 8B4D E0	MOV ECX,DWORD PTR SS:[LOCAL.8]	
0041E4A6	. BB 67000000	MOV EBX,67	
0041E4AB	. A1 78AA4900	MOV EAX,DWORD PTR DS:[49AA78]	
0041E4B0	. 89F2	MOV EDX,ESI	
0041E4B2	. 83F8 03	CMP EAX,3	
0041E4B5	. 75 36	JNE SHORT 0041E4ED	
0041E4B7	. B8 01000000	MOV EAX,1	
0041E4BC	> E8 77560000	CALL 00423B38	; [MDK95.00423B38
0041E4C1	. 8B4D E4	MOV ECX,DWORD PTR SS:[LOCAL.7]	
0041E4C4	. 89F2	MOV EDX,ESI	
0041E4C6	. 8B35 78AA4900	MOV ESI,DWORD PTR DS:[49AA78]	
0041E4CC	. BB 8B000000	MOV EBX,8B	
0041E4D1	. 83FE 04	CMP ESI,4	
0041E4D4	. 75 1B	JNE SHORT 0041E4F1	
0041E4D6	. B8 01000000	MOV EAX,1	
0041E4DB	. E8 58560000	CALL 00423B38	; [MDK95.00423B38
0041E4E0	. ^ E9 78FDFFFF	JMP 0041E25D	
0041E4E5	> 31C0	XOR EAX,EAX	
0041E4E7	. ^ EB 97	JMP SHORT 0041E480	
0041E4E9	> 31C0	XOR EAX,EAX	
0041E4EB	. ^ EB B1	JMP SHORT 0041E49E	
0041E4ED	> 31C0	XOR EAX,EAX	
0041E4EF	. ^ EB CB	JMP SHORT 0041E4BC	

```

0041E4F1 |> 31C0      XOR EAX,EAX
0041E4F3 |. E8 40560000 CALL 00423B38      ; [MDK95.00423B38
0041E4F8 \.^ E9 60FDFFFF JMP 0041E25D

```

In the patch, two loops are introduced which handle the loading and rendering. Furthermore, the rendering of the version number is introduced. The optimization produces a code pocket with enough space for the rest of the patch code. The following table shows the patched code including annotations.

	Patched	Comments
MOV CL,35		ASCII "5"
XOR ESI,ESI		ESI = 0 (used as current max width)
MOV EBX,OFFSET 004959EC		ASCII "OPT0"
MOV EDI,EBX		
ADD EDI,3		EDI points to number of OPT0 string
LEA EDX,[LOCAL.8]		EDX gets pointer to stack variable
MOV BYTE PTR DS:[EDI],CL		replace the OPT number with 5,4,3,2,1,0
MOV EAX,EBX		EAX points to OPT%d string
CALL GET_MDKFONT.FTI_ENTRY_BY_NAME(EAX)		find string in mdkfont.fti with id OPT%d
MOV DWORD PTR SS:[EDX],EAX		write the read text to stack variable
CALL 00414BE8		compute width of text in pixels
CMP ESI,EAX		check if width is >= max width
JGE SHORT 0041E343		jump if max width >= width
MOV ESI,EAX		set new max width as current width
SUB EDX,4		point to the next stack variable
DEC ECX		set CL to the next number (5,4,3,...)
CMP CL,30		check if CL reached the last number "0"
JGE SHORT 0041E32C		repeat loop if CL is not below "0"
MOV EDX,ESI		EDX = max width
SAR EDX,1F		
SUB ESI,EDX		the two commands produce abs(max width)
SAR ESI,1		divide abs(max width) by 2
MOV DWORD PTR SS:[LOCAL.14],ESI		store the computed value in stack variable
XOR ESI,ESI		ESI = 0
MOV ECX,DWORD PTR DS:[LASTGAME.sav exists?]		
TEST ECX,ECX		check if "Continue" needs to be rendered
LEA EDI,[LOCAL.13]		pointer to text with id "OPT0" on stack
JNZ SHORT 0041E36B		skip if there is a LASTGAME.sav
ADD EDI,4		get the next menu text entry
INC ESI		increase the running index
MOV EBX,1F		set EBX = 31
MOV ECX,DWORD PTR SS:[EDI]		grab the current menu item text
MOV EDX,DWORD PTR SS:[LOCAL.14]		get the (max width / 2) as center offset
MOV EAX,DWORD PTR DS:[Current_selected_menu_id]		grab the currently selected menu index
CMP EAX,ESI		check if current entry is the selected one
SETB BYTE PTR SS:[LOCAL.7]		activate the byte in stack var if it is
XOR EAX,EAX		clear EAX to 0
MOV AL,BYTE PTR SS:[LOCAL.7]		set EAX to the "is current selected" result
PUSH EBX		save y coord on stack
CALL 00423B38		render current menu entry
POP EBX		retrieve the y coord from stack
ADD EBX,24		increase y coord by 36
INC ESI		increase running index
ADD EDI,4		get the next menu text entry pointer
CMP ESI,5		check if all entries were rendered
JLE SHORT 0041E370		jump if not done yet
MOV EAX,0A		EAX = 10 used as color index
MOV EDX,230		EDX = 560 used as x coord
MOV EBX,160		EBX = 352 used as y coord
MOV ECX,0041E4F3		ASCII "v1.1"
CALL PRINT_F8_FONT_TEXT(EAX color, EDX X, EBX Y, ECX String)		render the version number on screen
JMP 0041E25D		continue with original code execution

## Change level table lookups to BYTE array instead of DWORD array

The following two snippets are added to the code pocket.

	Patched	Comments
0041E3B7	MOV EAX, <b>DWORD PTR DS:[Current_Level_Index]</b>	Assumption: the current level index is always <= 255
0041E3BC	MOV AL, <b>BYTE PTR DS:[EAX+Level_Index_LUT]</b>	Therefor, overwriting AL only produces a valid DWORD
0041E3C2	<b>RETN</b>	

	Patched	Comments
0041E3C3	XOR ECX, ECX	ECX = 0
0041E3C5	MOV CL, <b>BYTE PTR DS:[ESI+Level_Index_LUT]</b>	get the level number byte in CL
0041E3CB	<b>JMP 00420449</b>	

These snippets will be used at the following locations:

Original	Patched
0041B7D1 MOV EDI, <b>DWORD PTR DS:[EAX*4+Level_Index_LUT]</b>	0041B7D1 <b>CALL 0041E3B7</b> ; call the first snippet
...	0041B7D6 MOV EDI, EAX
...	...
00420442 MOV ECX, <b>DWORD PTR DS:[ESI*4+Level_Index_LUT]</b>	00420442 <b>JMP 0041E3C3</b> ; jump to the second snippet
...	...
0043395E MOV ECX, <b>DWORD PTR DS:[Current_Level_Index]</b>	0043395E <b>CALL 0041E3B7</b> ; call the first snippet
00433964 MOV EAX, <b>OFFSET 0054C5D0</b>	00433963 MOV ECX, EAX
00433969 XOR EDX, EDX	00433965 NOP
0043396B MOV ECX, <b>DWORD PTR DS:[ECX*4+Level_Index_LUT]</b>	00433966 NOP
	00433967 NOP
	00433968 NOP
	00433969 NOP
	0043396A NOP
	0043396B MOV EAX, <b>OFFSET 0054C5D0</b>
	00433970 XOR EDX, EDX

## Integrate the level selection menu entry in the main menu navigation

The index limit for the main menu needs to be extended to 5.

Original	Patched
0041DD2E MOV <b>DWORD PTR DS:[Current_selected_menu_id], 4</b>	0041DD2E MOV <b>DWORD PTR DS:[Current_selected_menu_id], 5</b>
...	...
0041DD6E <b>JL SHORT 0041DD87</b>	0041DD6E <b>JLE SHORT 0041DD87</b>
...	...
0041DDEA <b>JGE SHORT 0041DE16</b>	0041DDEA <b>JG SHORT 0041DE16</b>
...	...
0041DF26 <b>JGE 0041DE16</b>	0041DF26 <b>JG 0041DE16</b>

The jumps are adjusted to include the fifth menu entry. The level selection entry is placed as the fourth menu entry. This means that the "Options" and "Quit" entries need to be shifted by one. Furthermore, the level select will use the same handler function as the load game menu. This will be elaborated in a later chapter.

Original	Patched
0041DE71 <b>JNE 0041DFDF</b> ; jump to quit game, if not index 3	0041DE71 <b>JE 0041DFCB</b> ; trigger load menu handler
0041DE77 <b>CALL 00420CF0</b> ; trigger options menu handler	0041DE77 <b>JMP 0041E4B1</b> ; jump to code snippet below

The following code snippet is called, if the current menu index is greater three.

Patched	Comments
0041E4B1 CMP EDX, <b>4</b>	check if current menu index is 4
0041E4B4 <b>JNE 0041DFDF</b>	if not, jump to quit game as the menu index has to be 5 or higher
0041E4BA <b>CALL 00420CF0</b>	trigger options menu handler
0041E4BF <b>JMP 0041DE7C</b>	jumps to end of function

## Introduce the level selection as menu identifier 12 (0xC) and level select setup function

Each menu has a specific identifier which is used to call their respective menu handler functions. The menus have setup functions which will prepare necessary structures like a list of LBB files or list of saved games. Inside of these setup functions, the current active menu identifier gets set. The "Game State" for the main menu is 0, while the TRAVERSE section has a "Game State" of 3. The level select will only be called by the main menu which is why the function will only be extended for the "Game State" 0.

	Original	Patched
004202DA	CMP BYTE PTR DS:[Game_State],0	004202DA JMP 0041E436; go to the snippet below
004202E1	JNE 004204ED	

	Patched	Comments
0041E436	XOR EDI,EDI	EDI = 0 filter mask init as 0
0041E438	XOR ESI,ESI	ESI = 0 used to init the save game count to 0
0041E43A	CMP BYTE PTR DS:[Game_State],0	check from which Game State the menu is called
0041E441	JNE 004204ED	jump if not called from main menu
0041E447	CMP DWORD PTR DS:[Current_selected_menu_id],2	menu id 2 is Load Game entry
0041E44E	JE 004202E7	jump if load game was selected
0041E454	MOV DL,0C	level select was selected. Set DL to 12 (0xC)
0041E456	MOV EDI,10	File Attribute Constants: FILE_ATTRIBUTE_DIRECTORY 0x10
0041E45B	MOV EAX,OFFSET 004999F1; ASCII "SE\LEVEL*"	Use "SE\LEVEL*" instead of "*.sav"
0041E460	JMP 004202EE	continue execution at original code path

The code snippet not only sets the register DL to 12, which is later used to set the current active menu identifier, but is also sets the search mask and filter pattern for the level folders. In the find files function, the FILE\_ATTRIBUTE\_DIRECTORY = 16 is the mask to identify legitimate entries. For the level selection this means only folders shall be considered. Furthermore, the load game search string used "\*.sav" while the level select will search for folders with the pattern "LEVEL\*". The preceding "SE\" is part of the preceding "TRAVERSE" folder path and had to be added here due to limitations in the way the search path is created. This split can also be seen in the Data Segment entries mentioned above: "find path part 1" and "find path part 2".

Further changes in the menu setup function are given in the following table.

	Original	Patched
00420305	XOR EDX,EDX	00420305 MOV EDX,EDI; set EDX to given filter mask
...	...	...
0042030D	MOV DWORD PTR DS:[Savefiles_count],EDX	0042030D MOV DWORD PTR DS:[Savefiles_count],ESI; init to ESI(0)
...	...	...
00420305	XOR EDX,EDX	00420395 MOV EDX,EDI; set EDX to given filter mask

The setup function is responsible for creating the entry list which is displayed in the load game / level select menu. In the function which creates the search path, this additional patch is necessary to search for the entries in either the "SAVES" or "TRAVERSE" folder.

	Original	Patched
0041AB82	MOV ESI,OFFSET 00495390; ASCII "SAVES\"	0041AB82 JMP 0041E41E; jump to code snippet below

  

	Patched	Comments
0041E41E	MOV ESI,OFFSET 00495390; ASCII "SAVES\"	set the path to SAVES as in the original code
0041E423	CMP BYTE PTR DS:[Current_Menu_id],0C	check if the current menu is the level select
0041E42A	JNE SHORT 0041E431	if it is not level select, skip next line
0041E42C	MOV ESI,OFFSET 004999FB; ASCII "TRAVER"	use the part 2 of the search path instead
0041E431	JMP 0041AB87	jump back to the original code

Finally, the switch statement which handles the call of the respective menu handlers is adjusted to call the load game handler for the 0xC (12) entry as well. The load game menu handler has the id 0 in the switch statement.

	Original	Patched
004012EF	JA 00401167	004012EF JA 0041E40F; jump to code pocket below instead of quit game



	Patched	Comments
0041E40F	CMP AL, 0B	check if it is the level select id (0xb because the switch subtracts 1 before)
0041E411	JNE 00401167	if it is not the new level select id, continue to quit game (invalid id)
0041E417	MOV AL, 0	set the index to 0 which lets the switch statement call the load menu handler
0041E419	JMP 004012F5	jump back to the switch statement execution

### Reuse the load menu as level selection

The level select title text and "No Levels found" text need to be loaded for the 0xC menu. The following tables show the jumps to the code pockets which handle the differentiation between load game and level select menu.

	Original	Patched
00420BB0	MOV EAX, OFFSET 00495C7C; ASCII "SVOPT1"	00420BB0 JMP 0041E3DF
	...	...
00420CC8	MOV EAX, OFFSET 00495C74; ASCII "SVOPT3"	00420CC8 JMP 0041E3F7

	Patched	Comments
0041E3DF	CMP BYTE PTR DS:[Current_Menu_id], 0C	check if game is in level select menu
0041E3E6	MOV EAX, OFFSET 00495C7C	ASCII "SVOPT1" identifier of load game title text
0041E3EB	JNE SHORT 0041E3F2	skip next line if game is in load game menu
0041E3ED	MOV EAX, OFFSET 00499A02	ASCII "LT" identifier of level select title text
0041E3F2	JMP 00420BB5	jump back to the original code path

	Patched	Comments
0041E3F7	MOV EAX, OFFSET 00495C74	ASCII "SVOPT3" identifier of "No Save games found" text
0041E3FC	CMP BYTE PTR DS:[Current_Menu_id], 0C	check if game is in level select menu
0041E403	JNE SHORT 0041E40A	skip next line if game is in load game menu
0041E405	MOV EAX, OFFSET 00499A05	ASCII "LN" identifier of "No levels found" text
0041E40A	JMP 00420CCD	jump back to the original code path

The load game menu lists all .sav files present in the SAVES folder. For the level selection, the patch reuses this functionality. The save games also display either a screenshot of the save game location or the LBB for the level. The level selection will not display any image.

	Original	Patched
00420AD1	CMP EAX, DWORD PTR DS:[Current save image]	00420AD1 JMP 0041E465

	Patched	Comments
0041E465	CMP BYTE PTR DS:[Current_Menu_id], 0C	check if game is in level select
0041E46C	JE SHORT 0041E479	jump to clear image
0041E46E	CMP EAX, DWORD PTR DS:[Current save image]	original code operation
0041E474	JMP 00420AD7	return to original code
0041E479	CALL CLEAR_SCREENBUFFER_BLACK	clear the image
0041E47E	JMP 00420BA3	skip the save game image load / draw part

After selecting a level entry, the level number will be extracted from the item text.

	Original	Patched
0042095B	MOV EDX, DWORD PTR DS:[Savefiles_list]	0042095B JMP 0041E483

	Patched	Comments
0041E483	MOV EDX, DWORD PTR DS:[Savefiles_list]	get the level / savegame list in EDX
0041E489	ADD EAX, EDX	access the EAX-th entry of the list
0041E48B	CMP BYTE PTR DS:[Current_Menu_id], 0C	check if game is in level select
0041E492	JNE 00420963	if not in level select, continue original code
0041E498	ADD EAX, 5	skip over the "LEVEL" part of the entry
0041E49B	CALL ATOI	transform the text in EAX to a number
0041E4A0	PUSH EAX	save the number on stack
0041E4A1	CALL 0042056C	cleanup the level select menu structures
0041E4A6	POP EAX	retrieve the number from stack
0041E4A7	CALL 0041DC44	call the level load function
0041E4AC	JMP 004209A5	finish the function by jumping to end of it



If a level is selected, it needs to be loaded using its number. In the original game, with the use of debug options / cheats, the user can load levels by pressing the number keys. This function is used to load the levels. The game only stores the index of the current level and not the number itself. This means that only levels which are present in the level lookup table can be loaded. The patched code gets the number from the selected level folder and checks if it is part of the level lookup table. If not, the number will be put at the end of the level lookup list and the index set to it.

The original code still used the DWORD level lookup list while the patched one now uses the BYTE array.

Original	Patched
0041DC4A MOV ECX,EAX	0041DC4A XOR EDX,EDX
0041DC4C MOV EBX,DWORD PTR DS:[Level_Index_LUT]	0041DC4C CMP EDX,8
0041DC52 XOR EDX,EDX	0041DC4F JGE SHORT 0041DC5C
0041DC54 XOR EAX,EAX	0041DC51 CMP AL,BYTE PTR DS:[EDX+Level_Index_LUT]
0041DC56 CMP ECX,EBX	0041DC57 JE SHORT 0041DC6B
0041DC58 JE SHORT 0041DC6B	0041DC59 INC EDX
0041DC5A ADD EAX,4	0041DC5A JMP SHORT 0041DC4C
0041DC5D INC EDX	0041DC5C MOV BYTE PTR DS:[EDX+Level_Index_LUT],AL
0041DC5E CMP EAX,20	0041DC62 NOP
0041DC61 JGE SHORT 0041DC6B	0041DC63 NOP
0041DC63 CMP ECX,DWORD PTR DS:[EAX+Level_Index_LUT]	0041DC64 NOP
0041DC69 JNE SHORT 0041DC5A	0041DC65 NOP
	0041DC66 NOP
	0041DC67 NOP
	0041DC68 NOP
	0041DC69 NOP
	0041DC6A NOP
	0041DC6B

Finally, the last patch is related to the loading screen image. For the original levels, each level number has a respective LBB file. The custom levels will all resort to the newly added LOAD\_0.LBB. The following code will check if the level is part of the original level lookup table and otherwise limit the index to 0 which will load the custom LBB file. The patched code will also use the custom function for the level lookup as described in "Change level table lookups to BYTE array instead of DWORD array".

Original	Patched
00433D77 MOV ESI,DWORD PTR DS:[Current_Level_Index]	00433D77 CALL 0041E3D0
00433D7D MOV ESI,DWORD PTR DS:[ESI*4+Level_Index_LUT]	00433D7C NOP
00433D84 MOV EAX,OFFSET 0049756C; ASCII "LOAD_MSG"	00433D7D NOP
00433D89 PUSH ESI	00433D7E NOP
	00433D7F NOP
	00433D80 NOP
	00433D81 NOP
	00433D82 NOP
	00433D83 NOP
	00433D84 PUSH EAX
	00433D85 MOV EAX,OFFSET 0049756C; ASCII "LOAD_MSG"

Patched	Comments
0041E3D0 CALL 0041E3B7	call the level lookup to get the current level index
0041E3D5 MOV ESI,EAX	store the actual level index in ESI; this will be used to load the actual level
0041E3D7 CMP EAX,8	check if the index is one of the original levels
0041E3DA JBE SHORT 0041E3DE	skip next line if it is one of the original levels
0041E3DC XOR EAX,EAX	set the EAX to 0; this number will be used to load the LOAD_0.LBB
0041E3DE RETN	