PatchWork

Richard Körber

COLLABORATORS

	TITLE : PatchWork		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Richard Körber	January 13, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

PatchWork

1.1 PatchWork: Table Of Contents

#

# # # # # # #

#

#

##

#

#

V0.17

P R E R E L E A S E

ANYTHING IS SUBJECT TO CHANGE

_____

_____

-- TABLE OF CONTENTS --

_____

_____

PatchWork (C) 1997-2002 Richard Körber -- All Rights Reserved -- FreeWare

1.2   PatchWork: Introduction

I N T R O D U C T I O N

_____

_____

PatchWork is another system debugging tool like MuForce, Enforcer,

Mungwall, Poolwatch and so on. It also makes it's output to the

debug port, so you should use Sushi to read it, unless you have

got a second Amiga available...

Now, what does PatchWork do?

It all began when I read the exec.library autodocs and started to

wonder if all the warnings and restrictions mentioned in there are

really observed. There was no such tool available (at least I hope

so ;-), so I sat down and wrote a first test version of PatchWork.

I expected to see no single PatchWork hit, but I was wrong. There

are a lot of programs throwing PatchWork hits, recent programs as

well as some old Commodore commands! So there really seems to be a

demand for that tool...

Well, then I constructed a real tool out of PatchWork. I added

better outputs (as known from e.g. Enforcer) and checked even more

restrictions. The result is what you have on your harddisk now.

I stopped PatchWork due to the low acceptance of the community. Most

of the people didn't know about it after all, or just didn't use

it. Then Thomas Richter asked me to add PatchWork to his MuForce

packet. I agreed immediately, and resumed my work on PatchWork.

Anyhow, my time is very limited, so don't expect much more improvements,

unless there is a really overwhelming feedback and acceptance of the

_____

community. (Remember that you are a part of the community!)

Please note that this is only a pre-release. It is by far from

being complete, and I expect several bugs. But I want to know if

this tool is also useful for other programmers. If you like it,

please send me a small e-mail.

Remember: the future of PatchWork still depends on YOU!

Richard Körber

1.3   PatchWork: Features

F E A T U R E S

_____

_____

This are the features of PatchWork:

· Catches bad function calls from

- commodities.library

- dos.library

- exec.library

- graphics.library

- gadtools.library

- intuition.library

- utility.library

· Reminds you that a minimum OS version is required for special cases.

· Full debug output, including registers, stack, PC, disassembly and

SegTracker. As you are used from MuForce!

· Additionally, the function call will be shown, too!

· Four different severity levels, threshold adjustable

· Annoys you with pedantic PatchWork hits... ;-)

PatchWork is NOT:

· A replacement for Enforcer and/or Mungwall. Use them, too!

· A system protector that magically 'repairs' bad function calls.

This is not the philosophy behind PatchWork!

1.4   PatchWork: Copyright

C O P Y R I G H T

_____

_____

Please read the following parts carefully.

_____

If you do not entirely agree to these Copyright notes, you must delete

this archive and all related files.

_____

_____

COPYRIGHT

==========

NOTE: You accept the following terms by starting the software,

even for a test drive only.

PatchWork is Copyright (C) 1997-2002 Richard Körber.

All rights reserved.

You only have the right to use the software, but no rights on

the software itself. Disassembling, resourcing and all other ways

of reverse engineering is forbidden.

FREEWARE

PatchWork is FreeWare. You are allowed to use the packet without

paying a fee or similar to the author, or anybody else. So there is

no reason not to use it.

COPYING

You can copy the packet as long as it remains entire and unchanged.

You are allowed to compress the packet using a customary compression

software (as lha, lzh, lzx, dms). You must not compress single files of

the packet (e.g. PowerPacker or Imploder).

DISTRIBUTION

You must not exceed an usual price on the market for your working

and material. This means a maximum of 3 Euro (or the equivalent amount

in other currencies) for disks and 15 Euro for CD-ROMs containing a PD

software collection.

I explicitly permit the distribution via AmiNet series, and as part

of the mmu.library packet by Thomas Richter.

I explicitly forbid the distribution on a coverdisk of a magazine,

if it is announced as "Full Version", "Special Offer" or similar.

LIABILITY

You are using the program as it is, with all flaws, and on your

own risk! I grant no warranty for the software meeting a special

purpose. This software may cause financial damage or harm people.

Please note that one purpose of this software is to crash your

system if poor software is executed. This crash may also result in

data loss, loss of all unsaved documents and all files on your RAM

disk.

LIMITATIONS

You are not allowed to use this software and its results

· for fascist or military purposes

· on Amigas used in a sensible area

· if you do not agree to the copyright note

In this case you must delete the software and all related and

generated files immediately!

CONTENTS OF THE PACKAGE

The package is only entire with these files:

PatchWork/PatchWork (Main program)

PatchWork/PatchWork.guide (This documentation)

PatchWork.info

FILE_ID.DIZ

This package may be a part of the mmu.library package by Thomas

Richter.

1.5 PatchWork: Address

A D D R E S S

_____

_____

You can reach me through one of the following ways:

E MAIL

Use e-mails if ever possible. My address is:

amiga@shredzone.de

rkoeber@gmx.de

Please do not send mails larger than 40KB without prior permission.

If you should have serious trouble reaching me, check my web page for

other e-mail addresses.

SNAIL MAIL

If you do not have an e-mail account, you may also write a snail mail.

Please do NOT send any Amiga formatted disks. I am unable to read them

because my Amiga is defective. :-(

Richard Körber

Paffrather Straße 40

51465 Bergisch Gladbach

Germany

WORLD WIDE WEB

Updates are available through my web page. URL:

http://www.shredzone.de

1.6 PatchWork: Bugs

B U G S

_____

KNOWN BUGS

· PatchWork will hang up the system for some reason. I hope this bug

will be fixed when I implement the asynchronous I/O routines.

· There will for sure be some more bugs inside. Please help me find

them to make PatchWork a reliable tool.

KNOWN ENFORCER/MUNGWALL/PATCHWORK HITS

· none known yet... :-)

BUGGY SOFTWARE WORKAROUNDS

· If PatchWork crashes during startup, or shows an 8BADC0DE alert,

you should try the NOPERMIT option. Anyhow this is an indicator for

a flawed Patch Utility, so find out who made it and write him a

mail.

_____

PatchWork is written and tested on:

** HARDWARE **

System: Amiga 4000

CPU: CPU=68060/50 MHz (Rev 1), FPU=68060/50 MHz, MMU=68060

Chips: AGA (RAMSEY F, GARY Normal, CHUNKY None) VBR=0x0803AE08

Agnus: Alice 8374 Rev. 3-4 (Mode: PAL)

Denise: Lisa 8364 (Revision: 0)

AmigaOS: 3.5 (V40.68, BoingBag 1, SetPatch V44.6)

Exec V40.10 Workbench V44.2

Support: GraphicOS: CyberGraphX 4, AudioOS: AHI, TCP/IP: Miami

Clock: Power 50 Hz, VBlank 50 Hz, E 709379 Hz

RAM: Motherboard 32 bit, 60 ns, Double CAS, 4x Bandwidth

Memory: CHIP FAST TOTAL ROM = 512.0KB SLOW = 0

NORMAL ~2.0MB 42.0MB ~44.0MB

VIRT. 0 0 0

TOTAL ~2.0MB 42.0MB ~44.0MB

** EXPANSIONS **

Nr ID Adresse Größe Hersteller Produkt

----

1 2140.22 40000000 64MB Phase 5 CyberVision 64 Graphics

2 2140.19 00EA0000 128KB Phase 5 CyberStorm '060 MK-II Flash ROM

3 082C.10 00E90000 64KB BSC Multiface II Multi I/O

4 4754.0C 00EC0000 64KB MacroSystem Germany Toccata Audio

5 0877.CA 00ED0000 64KB VillageTronic Ariadne II Ethernet

6 4754.05 00EE0000 64KB MacroSystem Germany MaestroPro Audio

(This list has been made using Identify.)

1.7 PatchWork: History

H I S T O R Y

_____
_____

V0.17 » (14. 1.2002)

· Bugfix: "RunCommand() strlen(argptr) does not match to argsize"

complained though the strlen was correct in reality.

[Thomas Richter]

V0.16 » (5. 1.2000)

· PatchWork turned out to be millennium proof... ;-)

· "SetPointer() hot spot is outside of the sprite" is severity 0

now [Olivier Fabre convinced me]

V0.15 » (5.11.1999)

· V0.14 revision was not bumped

· Using disassembler.library now to disassemble around the PC,

with the parameters DISPC and DISRANGE (thanks, Thomas Richter)

V0.14 » (24.10.1999)

· gadtools.library tests implemented

· Documentation has been reworked

V0.13 » (10. 9.1999)

· Bugfix: CxBroker nb_Unique of 3 is legal [Jeroen T. Vermeulen]

· Ambiguous formulation in this guide's SetCxObjPri() chapter

[Jeroen T. Vermeulen]

· Added the NOPERMIT option

· Renamed "Badness" to "Severity"

· Reassembled it with proper optimizer settings, saving some

bytes

V0.12 » (4. 4.1998)

· Bugfix: the ShowHit function scratched A5, resulting that all

subsequent relative accesses to A5 threw enforcer hits. This

was a really ugly one. [Børge Nøst]

· Bugfix: Disable() implies Forbid(), this is honoured in all

exec checkings now. [Thomas Richter]

· Bugfix: BitMapScale width>1024 was wrong documented here. Anyhow

PatchWork was accurate. [Dave]

· DISABLECHECK implemented

· Permit() similar checking for Enable()

V0.11 » (12. 2.1998)

· utility.library tests implemented

· commodities.library tests implemented

V0.10 » (7.11.1997)

· Bugfix: BestModeIDA() said "0 is not allowed" when the tag is

exactly *non* null. No good... [Børge Nøst]

· Removed CopyMem() and CopyMemQuick() with 0 bytes hit. It was

quite annoying and is absolutely clean, anyhow.

· Added exec ReleaseSemaphore() and ReleaseSemaphoreList(). Don't

know why I left them out...

· AllocSpriteDataA() checks the SPRITEA_OldDataFormat tag now,

all further checkings are ignored then [Allan Purtle]

· DeleteMsgPort() now does a level 3 hit if port is still public

· Removed the superfluous and annoying CreateNewProc() suggestion

· intuition.library tests implemented

· PatchWork quits now if called a second time

V0.9 » (7.10.1997)

· Bugfix: signed compares for EraseRect() and RectFill()

· WaitBOVP() temporarily disabled, due to a clean usage in CGX3

· Added the MinOS option [Eric Sauvageau]

· Bad patch manager will result in an Alert (untested)

V0.8 » (6. 7.1997)

· AllocMem() and AllocVec() checking against -1 [Dave Jones]

V0.7 » (10. 6.1997)

· graphics.library tests implemented

· CopyMem() and CopyMemQuick() with overlapping areas are

finally Badness 3. Yeah!

· Automatical BPTR calculation

· NULL as STRPTR will not cause Enforcer hits any more

· WORD parameters are displayed correctly now

· Mistake in the guide file: AttemptLockDosList() returns

NULL for failure. PatchWork works accurately, though!

[Christian Wasner]

· CreateProc() isn't really obsoleted [Kenny], so it will

be just a suggestion to use CreateNewProc() and got severity

level 0.

V0.6 » (29. 5.1997)

· ROMHITS implemented

V0.5 » (29. 5.1997)

· dos.library tests implemented

V0.4 » (29. 5.1997)

· Improved output using a new structure, showing library

name, function parameters, severity level, cause

V0.3 » (28. 5.1997)

· STACKLINES,SHOWPC,STACKCHECK,AREGCHECK,DREGCHECK added

· Bugfix: bad Forbid() state recognition, works fine now

V0.2 » (28. 5.1997)

· CopyMem(), CopyMemQuick() react on size==0

· Permit() checks tdNestCnt

· DEADLY implemented

V0.1 » (26. 5.1997)

· First internal pre-release

1.8 PatchWork: Credits

C R E D I T S

_____

_____

_____

I wish to thank all the people who wrote me! :-)

I also want to credit:

Thomas Richter for MuForce, the disassembler.library and all his

kind help and hints. Without him, PatchWork would

have been discontinued.

Eric Sauvageau for his great MINOS idea!

Dave Jones for his help regarding memory flushing.

Frank Wille for PhxAss, the best assembler available! (FreeWare)

Michael Sinz for Enforcer, SegTracker and his help for all the

bugfree Amiga software.

———————————————————————————————————————
———————————————————————————————————

//

\\ // -- Amiga - Cow inside --

\X/

1.9 PatchWork: Future

F U T U R E

———————————————————————————————————————
———————————————————————————————————

This is what I plan in future releases:

· Limit the output to a task name using wildcards

· Using asynchronous output and also output on the shell.

· Fixing all bugs

· Checking even more functions and also other libraries

· Modularize it, and make the module format public so library programmers

can write PatchWork modules for their libraries.

If you have some more ideas, just write! :-)

Anyhow, I have a lot of other projects and very limited time, so please

be patient. Also, don't take anything for granted. Personally, I don't

believe that all the above will ever be implemented. (Sigh)

1.10 PatchWork: Usage

U S A G E

———————————————————————————————————————
———————————————————————————————————

REQUIREMENTS

———————————————————————————————————————

· AmigaOS 2.04 (V37) or higher (V36 won't work!)

· A terminal connected to the internal serial port, or Sushi

· SegTracker (not required, but suggested)

· disassembler.library by Thomas Richter (optionally)

INSTALLATION

Just copy PatchWork into your CLI path and this guide file into whereever

you store good documentations. ;)

You should start PatchWork after SetPatch and after any SetFunction()

enhancer (e.g. PatchControl).

START

You must start PatchWork from the shell.

PatchWork won't detach, so you should use "run <>NIL: PatchWork" if you

want to start it in the startup-sequence.

You can always remove PatchWork by pressing <CTRL>-<C>, sending a break,

or starting another instance of PatchWork.

SHELL PARAMETERS

LEVEL/K/N Set the severity threshold to be put out. Defaults

to 0. More about severity levels see below.

MINOS/K/N You can provide the minimal required OS version of your

program here. Any PatchWork hit that is fixed starting

from this OS version will be suppressed. E.g. if your

program requires at least AmigaOS 39, just provide MINOS=39.

TINY/S Tiny output. Just show the cause of the hit.

SMALL/S Small output. Shows the cause of the hit and the PC,

TCB and Task name.

STACKLINES/K/N Number of stack lines to be output. Each stack line

consists of 32 stack bytes. There is no limit here!

STACKCHECK/S Check also all stack entries with SegTracker

SHOWPC/S Show 32 bytes around the current PC

AREGCHECK/S Check also the contents of the address registers

with SegTracker.

DREGCHECK/S Check also the contents of the data registers with

SegTracker.

ROMHITS/S Usually, all hits caused by the AmigaOS ROM will be

suppressed. With this option set, they will also be shown.

DEADLY/S With this option set, some functions will return a worst

case scenario return code instead of the real return code

under certain conditions. See patch documentation below.

DISABLECHECK/S It is only allowed to disable system interrupts for a

maximum time of 250ms. This option will check if this time

range is kept. A 68020 or better is required for this test.

It won't work on most Amiga emulations and might trouble on

DraCo. Even more, this test could cause hardware timeouts,

so be very careful and do not use this option permanently.

If you don't know what I'm talking about, I strongly

encourage you not to use this option!

NOPERMIT/S Does not patch the Permit() function. This is useful

if your system crashes on start of PatchWork (or shows an

8BADC0DE alert), caused by a poor patch utility. There is no

reason to use this option if PatchWork starts up without any

problems.

DISPC/S Show a disassembly of the code around the current PC. This

option requires Thomas Richter's disassembler.library to

be installed properly.

DISRANGE/K/N Range of bytes to be disassembled before and after the

PC. Defaults to 32.

BADNESS LEVELS

All hits are divided into four levels of severity. The higher the

level, the more probable is a system crash caused by it.

0 Hits with level 0 won't crash the system. They are legal and

absolutely harmless. The only function of level 0 hits are

for mental notes, e.g. that a special OS version is required

for the function or the way it is used. Also see MINOS.

1 Hits with level 1 also won't crash the system, but they are

ugly, obsoleted, needless or waste system time, as e.g. an

"AllocMem(0)" or "OldOpenLibrary("...")".

2 Hits with level 2 might crash the system, especially with

multitasking interaction. This are e.g. FindPort()s without

a prior Forbid(). You must not rely on a result you get from

a level 2 hit! It has most probably already changed. If not,

it is just luck, but nothing more... ;-)

With the DEADLY option, some of the level 2 hits will try to

crash your program by returning worst case return codes.

3 Hits with level 3 will crash the system sooner or later. They

are used against the rules of AutoDocs, and thus will have

unpredictable results. Or they will result in memory loss. In

any way, they are used wrong.

Two examples: CopyMem() with overlapping memory areas, or

DeleteMsgPort() without clearing up the message queue before.

ANY hit starting from level 1 is a real bad hit, and should be

removed if possible, to enhance system stability and future compatibility.

1.11   PatchWork: How it works

H O W I T W O R K S

_____

_____

Piece of cake: PatchWork just patches some library functions.

If one of them is called, PatchWork checks the passed structures and

parameters and finds out if they meet the AutoDocs specifications.

If a test should fail, you will be notified through the debug

channel, or through the Sushi output console.

PatchWork is not made to protect the system from crashing. It won't

correct bad structures and parameters. It just reports what you forgot

and then leaves you alone with that mess.

In fact, you can even try to crash your program by using the DEADLY

option.

Note: we can discuss about anything, but not about something you

explicitly MUST or MUST NOT do regarding to the AutoDocs. The AutoDocs

are law! Even if it should work, it is simply wrong to break the rules,

and it will most probably fail in future releases! See the philosophy

chapter.

1.12   PatchWork: Patch Utilities

P A T C H U T I L I T I E S

_____

_____

The AmigaOS' patch function is quite limited. It is only possible to remove

all patches in the opposite order they were installed. Otherwise, trouble

will occur.

So, some smart programmers wrote utilities that patches the OS patch function

and replace it with a better one. Then you can remove all patches in any order
you like. I strongly suggest to use such a program.

Anyhow, some problems may occur, since PatchWork patches some fundamental
functions (e. g. AllocMem() and Permit()). If you have such problems, delete
the bad patch utility and find a new one.

STACK TROUBLE

It is crucial that PatchWork finds the caller's PC on the stack, and nothing
before that. If a patch utility pushes some data on the stack before starting
the patch functions, PatchWork will show the utilities PC instead of the callers
PC.

I currently don't know a patch utility with that problem. Anyhow, be prepared.

FUNCTION TROUBLE

A serious problem is if the new SetFunction() routine of the patch utility uses
any OS functions after setting the new function vector. Let me explain why.
For example, our program will patch the Permit() function.

This is the state before SetFunction() has been called.

```
+----------+ +--------------------+
BEFORE | Permit() |--------------------------->| Permit OS function |
+----------+ +--------------------+

+--------------+
| Permit patch |-----> NIL
+--------------+
```

We can see the Permit() entry pointing to the Permit OS function in ROM, and
our patch which has not yet been inserted.

Usually, SetFunction() writes the new function address into the library and
returns a pointer to the old function address.

```
After +----------+ +--------------------+
SetFunction() | Permit() |-+ Return --->| Permit OS function |
+----------+ | +--------------------+
| +--------------+
+-->| Permit patch |-----> NIL
+--------------+
```

Our patch can now jump to the original Permit OS function, thus building a
chain:

```
+----------+ +--------------------+
AFTER | Permit() |-+ +--->| Permit OS function |
+----------+ | | +--------------------+
```

| +-------------+ |

+-->| Permit patch |--+

+-------------+

But there is one weak point! After SetFunction() has been executed (but before

our patch points back to the ROM function), the Permit patch still points to NIL.

If someone would call Permit() at this stage, the system would crash immediately.

Usually, there is no chance for that, since patches are applied while the system

is Forbid()den. But when the patch utility calls Permit(), or breaks the disabled

state, the system will crash!

PatchWork does not point to NIL, but will show up the alert "8BADC0DE" instead.

If this one will show up while starting PatchWork, you will know that your

patch utility is faulty. Anyhow, this alert won't show in all cases. Another

good hint is if your system crashes after launching PatchWork.

WORKING ONES

These Utilities are known to work with PatchWork:

· PatchControl V2.3 or higher (AlienDesign, see MCP packet)

· SaferPatches V2.04 or higher

· No patch utility at all... ;-)

NOT WORKING ONES

These Utilities are known to fail:

· SaferPatches V2.03 and before

· MagicMenu V2.25 (use NOPERMIT)

INSTALLATION

If you are using patch utilities, then make sure that they are started before

PatchWork is running. Don't start them afterwards!

If the system will then crash during startup, or the alert occurs, then delete

that tool and use a newer or other one...

There is an alternative you may try: use the NOPERMIT parameter to disable the

exec/Permit() patch. This will help in some cases, as e.g. MagicMenu V2.25.

1.13 PatchWork: Output example

O U T P U T E X A M P L E

_____

_____

This is how a PatchWork hit looks like:

_____

exec.library OldOpenLibrary("dos.library")

Severity 1: obsoleted, use OpenLibrary() instead

PC=08339720 TCB=08399D58 ("Shell Process")

Data: 00000001 020CE5C5 00004000 08350BA4 00000001 00000001 020CFF85 08339714

Addr: 08350BA4 08339732 08014E0C 08339714 084F3C08 00F92D70 0800083C 084F3BFC

----> 08339732 - "pwtest" Hunk 0000, Offset 0000001A

----> 00F92D70 - "ROM - dos 39.23 (8.9.92)" Hunk 0000, Offset 00000314

PC-8: 00000000 00000000 0833653C 00000000 00000030 00000000 43FA0018 2C780004

PC *: 4EAEFE68 22402C78 00044EAE FE627000 4E75646F 732E6C69 62726172 79004E71

Stck: 084F3BFC 08339724 00F9359C 00004000 0839A74C 08533B40 00001970 48E7303E

Stck: 24482649 61A66730 2C6A0018 206A0014 4EAEFF94 2848204B 610000B8 661C2449

----> 08339720 - "pwtest" Hunk 0000, Offset 00000008

----> 08339724 - "pwtest" Hunk 0000, Offset 0000000C

----> 00F9359C - "ROM - dos 39.23 (8.9.92)" Hunk 0000, Offset 00000B40

_____

_____

Now for the single lines:

The first line contains what function caused the hit:

exec.library OldOpenLibrary("dos.library")

^^^^^^^^^^^^^--- passed parameters

^^^^^^^^^^^^^----------------- function name

^^^^^^^^^^^-------------------------------- library name

The second one describes the hit itself:

Severity 1: obsoleted, use OpenLibrary() instead

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^--- cause

^^^^^^^^^^--------------------------------------- severity level

(With the TINY option set, no more lines are displayed.)

The third line locates the hit:

PC=08339720 TCB=08399D58 ("Shell Process")

^^^^^^^^^^^^^^--- task name

^^^^^^^^-------------------- address of the task structure

^^^^^^^^-------------------------------- address of the JSR instruction

that caused the hit.

(With the SMALL option set, no more lines are displayed.)

After that, the data and address register contents follow:

Data: 00000001 020CE5C5 00004000 08350BA4 00000001 00000001 020CFF85 08339714

Addr: 08350BA4 08339732 08014E0C 08339714 084F3C08 00F92D70 0800083C 084F3BFC

If you enabled DREGCHECK or AREGCHECK, the contents of any register

will be passed to SegTracker (of course SegTracker must be running):

_____

----> 08339732 - "pwtest" Hunk 0000, Offset 0000001A

----> 00F92D70 - "ROM - dos 39.23 (8.9.92)" Hunk 0000, Offset 00000314

With SHOWPC enabled, 64 bytes around the PC at hit time will be shown:

PC-8: 00000000 00000000 0833653C 00000000 00000030 00000000 43FA0018 2C780004

PC *: 4EAEFE68 22402C78 00044EAE FE627000 4E75646F 732E6C69 62726172 79004E71

After that, all stack lines will be shown. You can change the number

of stack lines using the STACKLINES parameter. Defaults to 2 lines.

Stck: 084F3BFC 08339724 00F9359C 00004000 0839A74C 08533B40 00001970 48E7303E

Stck: 24482649 61A66730 2C6A0018 206A0014 4EAEFF94 2848204B 610000B8 661C2449

If SegTracker is running, it will be fed with the PC:

----> 08339720 - "pwtest" Hunk 0000, Offset 00000008

And last but not least, if STACKCHECK is enabled, all longwords shown

in the stack lines above are passed to SegTracker:

----> 08339724 - "pwtest" Hunk 0000, Offset 0000000C

----> 00F9359C - "ROM - dos 39.23 (8.9.92)" Hunk 0000, Offset 00000B40

1.14 PatchWork: Philosophy

P H I L O S O P H Y

_____

_____

I got a lot of feedback like: "Hey, this function is working fine, so

why does PatchWork bark about it?" Okay, let me tell you the philosophy

behind PatchWork, and software documentation in general.

In the early days, when the first home computers like the ZX Spectrum

and C64 were popular, computers were some kind of static. They were

constructed by some smart people, and then they were never touched again.

Even if there were hardware improvements, the kernel ROM did not change,

to ensure compatibility to the existing software.

This was because everyone could buy a book with the commented ROM

listing, and then they knew where to find what function, and exact details

of the kernel functionality. And since only one software could run on

these systems simultaneously, you didn't need to care about sharing

resources and you never had to deal with 3rd party software updates.

Well, times have changed. The static ROM has been replaced by operating

systems, which are updated frequently (at least they should be). These

systems are able to run a number of programs simultaneously, but since

they run on the same hardware, they have to share their resources.

You have to follow exact rules if you want to write software that can

interact with several versions of the operating system and other software installed on the software.

First, you always have to use the operating system instead of accessing the hardware resources yourself. A good example for this is the DraCo, which is able to run all conformous software without modifications, just because they do not access the Amiga chipset directly.

The other, even more crucial thing is documentation. You don't know how the operating system or the piece of software works exactly. You aren't ought to know it, and you won't even be able to know since software is growing more and more complex. So there is only one interface between your software and the operating system. This is the documentation that explains how your operating system works.

If you are a tool programmer, you rely that the OS call works exactly as documented. You want to have a certain result if you pass in the right mixture of parameters.

If you are a driver or patch programmer, you rely that OS functions were called in a definite way. You cannot catch all variations and eventualities, since this would unnecessarily slow down your function.

I want to mention a good example of what happens if people ignore the documentation. When Amiga started to get popular, it was delivered with the AmigaOS 1.2, and later with AmigaOS 1.3. Some people wrote programs like they have been used to write it on their C64: they knew the address of some functions in the ROM, and they knew for example that the graphics.library was always the 3rd library after exec.library (or was it the 4th, I don't know exactly). So, to save some microseconds, they ignored the documentation and used their hacks. And they worked fine!

Until the day when Commodore released the AmigaOS 2.0.

This was really a black day for the Amiga, because a LOT of games and hacker tools crashed on the new AmigaOS. Commodore was not to blame, they did the right thing. Instead of trying to be compatible to as much existing software as possible (which was the philosophy of Microsoft these days), they were only downward compatible to their documented features. So all conformous software still worked on AmigaOS 2.0, while all the hacks crashed awfully.

Unfortunately, Commodore WAS blamed for the crashing applications! In all shops, the new Amiga models were considered as unstable, not compatible and just awful. I heard a lot of vendors who told their customers this scrap! They suggested to buy the AmigaOS 1.3 and a switchboard as well, but the mounting would void the warranty. Most customers decided to better

buy a PC with Windows 3.1, which was said to be more stable.

This rumour of AmigaOS being incompatible and unstable, is told until

today. And all this just because some lame programmers ignored the rules

and wrote some bad hacks.

As you see, the documentation is the most important part if you want to

write rock stable and reliable software. And this is why you should use

PatchWork! Certainly, PatchWork is no guarantee that your software will

still run on future AmigaOS releases, but it comes very close to that.

PatchWork knows about the AutoDocs, the MUSTs and MUST-NOTs. And it is

quite pedantic. It will surely annoy you some time with hits that seem to

be harmless at the first sight. But if you fix these hits, you can be sure

that your program might also work with clean system patches, OS updates or

other software.

Anyhow, if you don't get any PatchWork hits, it is no sign for your

program being 100% compatible. It also depends on some other factors. The

best rule of thumb is not to play the smart Alec and regard undocumented

features as given. If you find some behaviour that is not documented,

better try to find a documented way to reach your aim, instead of just

using the undocumented feature and rely on that it will still work in

future versions.

1.15 PatchWork: commodities.library

c o m m o d i t i e s . l i b r a r y

_____

_____

These commodities.library patches are implemented:

AttachCxObj

CxBroker

CxMsgData

CxMsgID

CxMsgType

DisposeCxMsg

DivertCxMsg

EnqueueCxObj

InsertCxObj

RouteCxMsg

SetCxObjPri

1.16   PatchWork: commodities.library / AttachCxObj

c o m m o d i t i e s . l i b r a r y AttachCxObj

_____

REPORT V38+ required for headObj==NULL

SEVERITY Level 0, MinOS 38

EXPLANATION NULL does only work since V38

DEADLY has no effect.

1.17   PatchWork: commodities.library / CxBroker

c o m m o d i t i e s . l i b r a r y CxBroker

_____

REPORT nb_Name string is too long

nb_Title string is too long

nb_Descr string is too long

SEVERITY Level 3

EXPLANATION You must keep the maximum string length as defined in

<libraries/commodities.h>. If you use locale.library for

the strings, set the maximum length in the .cd as well.

DEADLY has no effect.

_____

REPORT nb_Name string is NULL

nb_Title string is NULL

nb_Descr string is NULL

SEVERITY Level 3

EXPLANATION You must provide these strings.

DEADLY has no effect.

_____

REPORT nb_Unique 1234 is not allowed

SEVERITY Level 1

EXPLANATION The provided nb_Unique value is not defined in the

<libraries/commodities.h>. Do not forget to initialize

this variable as well!

DEADLY has no effect.

_____

1.18 PatchWork: commodities.library / CxMsgData

c o m m o d i t i e s . l i b r a r y CxMsgData

_____

_____

REPORT V38+ required for cxm==NULL

SEVERITY Level 0, MinOS 38

EXPLANATION NULL does only work since V38

DEADLY has no effect.

1.19 PatchWork: commodities.library / CxMsgID

c o m m o d i t i e s . l i b r a r y CxMsgID

_____

_____

REPORT cxm must not be NULL

SEVERITY Level 3

EXPLANATION NULL is not allowed, and will throw Enforcer hits.

DEADLY has no effect.

1.20 PatchWork: commodities.library / CxMsgType

c o m m o d i t i e s . l i b r a r y CxMsgType

_____

_____

REPORT cxm must not be NULL

SEVERITY Level 3

EXPLANATION NULL is not allowed, and will throw Enforcer hits.

DEADLY has no effect.

1.21 PatchWork: commodities.library / DisposeCxMsg

c o m m o d i t i e s . l i b r a r y DisposeCxMsg

_____

_____

REPORT cxm must not be NULL

SEVERITY Level 3

EXPLANATION NULL is not allowed, and will throw Enforcer hits.

DEADLY has no effect.

1.22   PatchWork: commodities.library / DivertCxMsg

c o m m o d i t i e s . l i b r a r y DivertCxMsg

_____

_____

REPORT cxm must not be NULL

SEVERITY Level 3

EXPLANATION NULL is not allowed, and will throw Enforcer hits.

DEADLY has no effect.

1.23   PatchWork: commodities.library / EnqueueCxObj

c o m m o d i t i e s . l i b r a r y EnqueueCxObj

_____

_____

REPORT V38+ required for headObj==NULL

SEVERITY Level 0, MinOS 38

EXPLANATION NULL does only work since V38

DEADLY has no effect.

1.24   PatchWork: commodities.library / InsertCxObj

c o m m o d i t i e s . l i b r a r y InsertCxObj

_____

_____

REPORT V38+ required for headObj==NULL

SEVERITY Level 0, MinOS 38

EXPLANATION NULL does only work since V38

DEADLY has no effect.

1.25   PatchWork: commodities.library / RouteCxMsg

c o m m o d i t i e s . l i b r a r y RouteCxMsg

_____

_____

REPORT cxm must not be NULL

SEVERITY Level 3

EXPLANATION NULL is not allowed, and will throw Enforcer hits.

DEADLY has no effect.

REPORT co must not be NULL

SEVERITY Level 3

EXPLANATION NULL is not allowed, and will throw Enforcer hits.

DEADLY has no effect.

1.26   PatchWork: commodities.library / SetCxObjPri

c o m m o d i t i e s . l i b r a r y SetCxObjPri

REPORT Priority is out of range (-128..127)

SEVERITY Level 1

EXPLANATION Even though the parameter is LONG, you must keep the

BYTE range as stated in the AutoDocs.

DEADLY has no effect.

REPORT V38+ required for a result

SEVERITY Level 0, MinOS 38

EXPLANATION The old CxObj priority is only returned since V38.

DEADLY 0xFACEDEAD is returned in any case.

1.27   PatchWork: dos.library

d o s . l i b r a r y

These dos.library patches are implemented:

AttemptLockDosList

CreateProc

DoPkt

Examine

ExamineFH

ExAll

ExAllEnd

ExNext

GetVar

Info

MatchEnd

NOTE: BPTRs are marked with a 'Bx' prefix and will

automagically be transformed into an APTR for your convenience!

1.28 PatchWork: dos.library / AttemptLockDosList

d o s . l i b r a r y AttemptLockDosList

_____

_____

REPORT also returns 0x00000001 until V39.24 dos

SEVERITY Level 0, MinOS 40

EXPLANATION Due to a bug, this function could also return 1

for failure. This bug is fixed in dos.library V39.24,

which was released after AmigaOS V39.106! Thus,

a check against dos V39 is not sufficient.

I hope you took this into consideration when using the

function!

DEADLY If not successful, DEADLY returns the worst case

return code, namely 0x00000001. If your program is bad, it

will think about a successful call even though this call

failed. Unfortunately, this won't crash your system. :(

1.29 PatchWork: dos.library / CreateProc

d o s . l i b r a r y CreateProc

_____

_____

REPORT pri is out of range (-128..127)

SEVERITY Level 2

EXPLANATION Even though pri is LONG, you must keep a BYTE

range!

DEADLY has no effect.

_____

REPORT stack size must be a multiple of 4

SEVERITY Level 2

EXPLANATION No further comment required, I think...

DEADLY has no effect.

_____

1.30   PatchWork: dos.library / DoPkt

d o s . l i b r a r y DoPkt

_____

REPORT DoPkt() from a task requires V37+

SEVERITY Level 0, MinOS 37

EXPLANATION You called DoPkt() from a task. This is only possible

since AmigaOS V37. Remember that!

DEADLY has no effect.

1.31   PatchWork: dos.library / Examine

d o s . l i b r a r y Examine

_____

REPORT FileInfoBlock is not longword aligned

SEVERITY Level 3

EXPLANATION Longword alignment is vital for this function. Other-

wise Enforcer and Mungwall hits will occur! Creating this

structure on stack is not sufficient.

DEADLY has no effect.

1.32   PatchWork: dos.library / ExamineFH

d o s . l i b r a r y ExamineFH

_____

REPORT FileInfoBlock is not longword aligned

SEVERITY Level 3

EXPLANATION Longword alignment is vital for this function. Other-

wise Enforcer and Mungwall hits will occur! Creating this

structure on stack is not sufficient.

DEADLY has no effect.

1.33   PatchWork: dos.library / ExAll

d o s . l i b r a r y ExAll

_____

REPORT buffer is not word aligned

SEVERITY Level 3

EXPLANATION buffer must be at least word aligned. Better do a

longword alignment if possible!

DEADLY has no effect.

_____

REPORT buffer should be longword aligned

SEVERITY Level 1

EXPLANATION It is even better if you longword align the buffer.

And it doesn't hurt. So why don't you do it? :-)

DEADLY has no effect.

1.34   PatchWork: dos.library / ExAllEnd

d o s . l i b r a r y ExAllEnd

_____

REPORT buffer is not word aligned

SEVERITY Level 3

EXPLANATION buffer must be at least word aligned. Do a longword

alignment if ever possible!

DEADLY has no effect.

_____

REPORT buffer should be longword aligned

SEVERITY Level 1

EXPLANATION It is even better if you longword align the buffer.

And it doesn't hurt. So why don't you do it? :-)

DEADLY has no effect.

1.35   PatchWork: dos.library / ExNext

d o s . l i b r a r y ExNext

_____

_____

REPORT FileInfoBlock is not longword aligned

SEVERITY Level 3

EXPLANATION Longword alignment is vital for this function. Other-

wise Enforcer and Mungwall hits will occur! Creating this

structure on stack is not sufficient.

DEADLY has no effect.

1.36 PatchWork: dos.library / GetVar

d o s . l i b r a r y GetVar

_____

REPORT this flags will require V37+

SEVERITY Level 0, MinOS 37

EXPLANATION GVF_DONT_NULL_TERM in combination with GVF_LOCAL_ONLY

is only available since AmigaOS V37.

DEADLY has no effect.

_____

REPORT this flags will require V39+

SEVERITY Level 0, MinOS 39

EXPLANATION GVF_DONT_NULL_TERM without GVF_LOCAL_ONLY is only

available since AmigaOS V39.

DEADLY has no effect.

1.37 PatchWork: dos.library / Info

d o s . l i b r a r y Info

_____

REPORT parameterBlock is not longword aligned

SEVERITY Level 3

EXPLANATION Longword alignment is vital for this function. Other-

wise Enforcer and Mungwall hits will occur! Creating this

structure on stack is not sufficient.

DEADLY has no effect.

1.38 PatchWork: dos.library / MatchEnd

d o s . l i b r a r y MatchEnd

_____

_____

REPORT AnchorPath is not longword aligned

SEVERITY Level 3

EXPLANATION Longword alignment is vital for this function. Other-

wise Enforcer and Mungwall hits will occur! Creating this

structure on stack is not sufficient.

DEADLY has no effect.

1.39 PatchWork: dos.library / MatchFirst

d o s . l i b r a r y MatchFirst

_____

_____

REPORT AnchorPath is not longword aligned

SEVERITY Level 3

EXPLANATION Longword alignment is vital for this function. Other-

wise Enforcer and Mungwall hits will occur! Creating this

structure on stack is not sufficient.

DEADLY has no effect.

REMARK In fact, it is quite funny that even Commodore did it

wrong on AddDataTypes, Dir and Mount. Use "Patches" from

Flavio Stanchina (flavio@ies.it) to remove those hits.

1.40 PatchWork: dos.library / MatchNext

d o s . l i b r a r y MatchNext

_____

_____

REPORT AnchorPath is not longword aligned

SEVERITY Level 3

EXPLANATION Longword alignment is vital for this function. Other-

wise Enforcer and Mungwall hits will occur! Creating this

structure on stack is not sufficient.

DEADLY has no effect.

_____

1.41   PatchWork: dos.library / RunCommand

d o s . l i b r a r y RunCommand

_____

_____

REPORT argptr does not end with '\n'

SEVERITY Level 2, MinOS 38

EXPLANATION The argument string must end with \n, so even old parsers

and old AmigaOS releases can handle it correctly.

DEADLY has no effect.

_____

REPORT strlen(argptr) does not match to argsize

SEVERITY Level 1

EXPLANATION argsize does not match the real argument string length.

DEADLY has no effect.

1.42   PatchWork: dos.library / SetVBuf

d o s . l i b r a r y SetVBuf

_____

_____

REPORT buff is not longword aligned

SEVERITY Level 3

EXPLANATION You must longword align the buffer!

DEADLY has no effect.

_____

REPORT not implemented until V40+

SEVERITY Level 0, MinOS 40

EXPLANATION In fact, this function is not implemented up to AmigaOS

V39! You can call it, but it will always succeed with

returncode 0, even though it did nothing!

DEADLY has no effect.

1.43   PatchWork: exec.library

e x e c . l i b r a r y

_____

_____

These exec.library patches are implemented:

_____

AddPort

AllocMem

AllocVec

CopyMem

CopyMemQuick

CreateIORequest

DeleteMsgPort

Enable

FindPort

FindSemaphore

FindTask

FreeSignal

InitSemaphore

OldOpenLibrary

Permit

Procure

ReleaseSemaphore

ReleaseSemaphoreList

SetFunction

Vacate

1.44 PatchWork: exec.library / AddPort

e x e c . l i b r a r y AddPort

_____

_____

REPORT port name not initialized

SEVERITY Level 2

EXPLANATION With this function, the MsgPort will be made publically

available. Thus, a valid ln_Name entry must be provided,

so FindPort() can find your node without trouble.

If ln_Name is a NULL pointer, FindPort() will cause

Enforcer hits.

DEADLY has no effect.

1.45   PatchWork: exec.library / AllocMem

e x e c . l i b r a r y AllocMem

———

——

REPORT allocating 0 bytes

SEVERITY Level 1

EXPLANATION Allocating 0 bytes is ugly, wastes time and might also

fail. Avoid this if possible.

DEADLY has no effect.

———

REPORT don't use AllocMem(-1) to flush memory

SEVERITY Level 1

EXPLANATION To flush the memory, you should either call AvailMem(),

then add some byte to the result and allocate this, or (not

that good, but still acceptable) do an AllocVec(0x7FFFFFF0).

DEADLY has no effect.

1.46   PatchWork: exec.library / AllocVec

e x e c . l i b r a r y AllocVec

———

——

REPORT allocating 0 bytes

SEVERITY Level 1

EXPLANATION Allocating 0 bytes is ugly, wastes time and might also

fail. Avoid this if possible.

DEADLY has no effect.

———

REPORT don't use AllocVec(-1) to flush memory

SEVERITY Level 1

EXPLANATION To flush the memory, you should either call AvailMem(),

then add some byte to the result and allocate this, or (not

that good, but still acceptable) do an AllocVec(0x7FFFFFF0).

DEADLY has no effect.

1.47   PatchWork: exec.library / CopyMem

e x e c . l i b r a r y CopyMem

_____

REPORT memory areas are overlapping (incremental)

SEVERITY Level 3

EXPLANATION Copying overlapping memory areas are not supported. See

AutoDocs! This is a incremental copy (destination before

source).

DEADLY has no effect.

REMARK I got a lot of request to remove this hit, or at least

set it to Level 1 or so because it works fine. This is

against the philosophy of PatchWork. To cite the AutoDocs:

"Arbitrary overlapping copies are not supported."

I think this is plain speak. We have to obey this, if we

like it or not, and if it may currently work or not.

I will remove this hit as soon as the AutoDocs says that it

is all right to do so. No further discussion.

_____

REPORT memory areas are overlapping (decremental)

SEVERITY Level 3

EXPLANATION Copying overlapping memory areas is not supported. See

AutoDocs! This is a decremental copy (source before

destination), which will even fail on AmigaOS up to 3.1!

DEADLY has no effect.

1.48   PatchWork: exec.library / CopyMemQuick

e x e c . l i b r a r y CopyMemQuick

_____

REPORT pointer/size not longword aligned

SEVERITY Level 3

EXPLANATION This is a raw violation, and will most probably cause

a crash! In this case, you MUST use CopyMem() instead.

DEADLY has no effect.

_____

REPORT memory areas are overlapping (incremental)

_____

SEVERITY Level 3

EXPLANATION Copying overlapping memory areas are not supported. See

AutoDocs! This is a incremental copy (destination before

source).

DEADLY has no effect.

REMARK I got a lot of request to remove this hit, or at least

set it to Level 1 or so because it works fine. This is

against the philosophy of PatchWork. To cite the AutoDocs:

"Arbitrary overlapping copies are not supported."

I think this is plain speak. We have to obey this, if we

like it or not, and if it may currently work or not.

I will remove this hit as soon as the AutoDocs says that it

is all right to do so. No further discussion.

——

REPORT memory areas are overlapping (decremental)

SEVERITY Level 3

EXPLANATION Copying overlapping memory areas is not supported. See

AutoDocs! This is a decremental copy (source before

destination), which will even fail on AmigaOS up to 3.1!

DEADLY has no effect.

1.49   PatchWork: exec.library / CreateIORequest

e x e c . l i b r a r y CreateIORequest

——

——

REPORT size is too small

SEVERITY Level 3

EXPLANATION There is something wrong with your code. Size cannot

be smaller than the size of a minimal IORequest struc-

ture. This will most probably cause Enforcer hits and

Mungwall hits.

DEADLY has no effect.

——

REPORT ioReplyPort not initialized

SEVERITY Level 2

EXPLANATION CreateIORequest requires an initialized structure. See

AutoDocs!

DEADLY has no effect.

——

1.50   PatchWork: exec.library / DeleteMsgPort

e x e c . l i b r a r y DeleteMsgPort

_____

_____

REPORT MsgPort contained unreplied Messages

SEVERITY Level 3

EXPLANATION You must reply all messages you have in your MsgPort

before deleting it. Otherwise it will cause memory loss

and some programs might wait for the reply forever. Also

see AutoDocs!

DEADLY has no effect.

_____

REPORT MsgPort is still public

SEVERITY Level 3

EXPLANATION First call RemPort() before deleting the port!

DEADLY has no effect.

1.51   PatchWork: exec.library / Enable

e x e c . l i b r a r y Enable

_____

_____

REPORT Disable() missing

SEVERITY Level 2

EXPLANATION There was an Enable() call though interrupts are already

enabled. Do not use Disable() to neutralize a following

Enable().

DEADLY has no effect.

_____

REPORT Disable time exceeded (1234 ms)

SEVERITY Level 3

EXPLANATION Only if DISABLECHECK has been enabled!

Autodocs allow to disable interrupts for a maximum of

250ms. This time has been exceeded. Try to split up the

code in the disabled section, so interrupts are enabled

in appropriate time slices. It would be even better if

you find a solution to avoid disabling the interrupts

at all.

_____

This test requires at least an 68020, and will not work

properly on Amiga emulations.

It may also occur if the disabled section threw other

hits.

DEADLY has no effect.

1.52 PatchWork: exec.library / FindPort

e x e c . l i b r a r y FindPort

_____

_____

REPORT Forbid() missing, unreliable result

SEVERITY Level 2

EXPLANATION This is one of the hits that occures most. You MUST

freeze multitasking before searching through the public

list for a message port. Otherwise the port could be

shut up shortly after you've got the pointer. The next

PutMsg() will then throw Enforcer hits in best case, or

it will just trash memory. Also see AutoDocs!

DEADLY In DEADLY mode, this function call will return the

true return code only if multitasking was forbidden.

Otherwise it will pass NULL if the Port wasn't there

anyhow, or 0xFACEDEAD if the Port was found. So, just

finding out if this port existed will still work, but

accessing it will cause Enforcer hits.

REMARK Some people claim that this hit is annoying, since they

consider this practice to be legal if you only want to

find out if a port is available or not.

I don't think so. There is still a slight chance to

crash the system. Imagine your program is iterating the

list without Forbid(), looking for a certain Node B

which comes after Node A. Now, while FindPort() is

fetching Node A, a scheduling occurs and another process

gets the processor. This other process now removes Node A

from the list and changes its memory contents. If your

program then gets the processor back, it will get a

completely messed up 'Node A', and will crash.

It is sad, but as long as AmigaOS does not provide a

semaphore mechanism for public lists, we must use Forbid()

if we iterate through them.

1.53   PatchWork: exec.library / FindSemaphore

e x e c . l i b r a r y FindSemaphore

_____

REPORT Forbid() missing, unreliable result

SEVERITY Level 2

EXPLANATION You MUST freeze multitasking before searching a

public Semaphore. Otherwise the semaphore could be

removed shortly after you've got the pointer. An access

to the structure will then throw Enforcer hits in best

case, or it will just trash memory. Also see AutoDocs!

DEADLY In DEADLY mode, this function call will return the

true return code only if multitasking was forbidden.

Otherwise it will pass NULL if the Semaphore wasn't

there anyhow, or 0xFACEDEAD if the Semaphore was found.

So, just finding out if this semaphore existed will still

work, but accessing it will cause Enforcer hits.

REMARK Some people claim that this hit is annoying, since they

consider this practice to be legal if you only want to

find out if a port is available or not.

I don't think so. There is still a slight chance to

crash the system. Imagine your program is iterating the

list without Forbid(), looking for a certain Node B

which comes after Node A. Now, while FindPort() is

fetching Node A, a scheduling occurs and another process

gets the processor. This other process now removes Node A

from the list and changes its memory contents. If your

program then gets the processor back, it will get a

completely messed up 'Node A', and will crash.

It is sad, but as long as AmigaOS does not provide a

semaphore mechanism for public lists, we must use Forbid()

if we iterate through them.

1.54   PatchWork: exec.library / FindTask

e x e c . l i b r a r y FindTask

_____

REPORT Forbid() missing, unreliable result

SEVERITY Level 2

EXPLANATION If you search for a task by its name, you MUST freeze
multitasking before searching through the public task
list. Otherwise the task could be removed shortly after
you've got the pointer. Any access to the TCB will then
throw Enforcer hits in best case, or it will just trash
memory.

Anyhow, FindTask(NULL) will work fine even with enabled
multitasking. This is because there is no iterating
through the public lists of tasks required to find the
own task, and if you can use the result, you'll surely
be still alive... ;-)

DEADLY In DEADLY mode, this function call will return the
true return code only if multitasking was forbidden
or NULL has been passed as task name. Otherwise it will
return NULL if the task really wasn't there, or
0xFACEDEAD if the task was found. So, just finding out
if this task existed will still work, but accessing it
will cause Enforcer hits.

REMARK Some people claim that this hit is annoying, since they
consider this practice to be legal if you only want to
find out if a port is available or not.
I don't think so. There is still a slight chance to
crash the system. Imagine your program is iterating the
list without Forbid(), looking for a certain Node B
which comes after Node A. Now, while FindPort() is
fetching Node A, a scheduling occurs and another process
gets the processor. This other process now removes Node A
from the list and changes its memory contents. If your
program then gets the processor back, it will get a
completely messed up 'Node A', and will crash.
It is sad, but as long as AmigaOS does not provide a
semaphore mechanism for public lists, we must use Forbid()
if we iterate through them.

1.55   PatchWork: exec.library / FreeSignal

e x e c . l i b r a r y FreeSignal

_____

REPORT V37+ will be required

SEVERITY Level 0, MinOS 37

EXPLANATION Just a rememberance that you'll need at least AmigaOS

V37 to call FreeSignal(-1).

DEADLY has no effect.

1.56   PatchWork: exec.library / InitSemaphore

e x e c . l i b r a r y InitSemaphore

_____

REPORT structure is not cleared

SEVERITY Level 2

EXPLANATION The memory space passed to be initialized as a new

semaphore must be cleared according to the AutoDocs.

Even if it seemingly works fine without clearing up the

memory mess, you must do it! No further discussion.

DEADLY has no effect.

1.57   PatchWork: exec.library / OldOpenLibrary

e x e c . l i b r a r y OldOpenLibrary

_____

REPORT obsoleted, use OpenLibrary() instead

SEVERITY Level 1

EXPLANATION Not much to say. Just follow the advice! But please

do NOT just replace it by a lazy OpenLibrary(...,0).

This isn't much better...

DEADLY has no effect. Yet...

REMARK I've got a lot of response on this one. All of them have

two arguments in common that people gave against this hit.

The first one is that the AutoDocs do not explicitly say

not to use this function. Anyhow, this is implied by: "This

obsolete function is provided so that object code compiled

using a 1.0 system will still run." To say it in another

words: this call is only to be used in software that is

ought to run under Kick 1.0. Obsolete functions must not be

used in new code!

The second argument is that you can save 2 (in words: two)

byte by using this function instead of OpenLibrary(). This

is ridiculous, IMHO. First of all, let your program have

10 Open calls (which is rather much), you'd save 20 byte.

This is no price for future compatibility. Another problem

is the missing library version checking.

1.58   PatchWork: exec.library / Permit

e x e c . l i b r a r y Permit

_____

_____

REPORT Forbid() missing

SEVERITY Level 2

EXPLANATION There was a Permit() call though multitasking is already

active. Do not use Permit() to neutralize a following

Forbid().

DEADLY has no effect.

REMARK This check will not be performed with the NOPERMIT

option set.

1.59   PatchWork: exec.library / Procure

e x e c . l i b r a r y Procure

_____

_____

REPORT V39+ will be required

SEVERITY Level 0, MinOS 39

EXPLANATION Procure() is only working in V39 or higher. Don't use

it on older AmigaOS releases!

DEADLY has no effect.

1.60 PatchWork: exec.library / ReleaseSemaphore

e x e c . l i b r a r y ReleaseSemaphore

_____
_____

REPORT semaphore is not obtained

SEVERITY Level 3

EXPLANATION The semaphore has already been released. You must

release any semaphore exactly as often as you obtained

it.

DEADLY has no effect.

1.61 PatchWork: exec.library / ReleaseSemaphoreList

e x e c . l i b r a r y ReleaseSemaphoreList

_____
_____

REPORT semaphore @0x4321ABDC is not obtained

SEVERITY Level 3

EXPLANATION The semaphore has already been released. You must

release any semaphore exactly as often as you obtained

it.

DEADLY has no effect.

1.62 PatchWork: exec.library / SetFunction

e x e c . l i b r a r y SetFunction

_____
_____

REPORT The function offset must be negative

SEVERITY Level 3

EXPLANATION You must provide a negative function offset!

DEADLY has no effect.

_____

REPORT Requested function does not exist

SEVERITY Level 3

EXPLANATION The library does not have a function with that offset. If

you want to patch functions which are only available in

later OS versions, you must make sure that you have the

_____

right version number before applying the patch!

DEADLY has no effect.

REPORT Your patch is called before SetFunction() returns

SEVERITY Level 2

EXPLANATION This is a lack of the AutoDocs, so be careful. If you use

the SetFunction()'s result to build a chain, you must not

patch CacheClearU() and SumLibrary() since SetFunction()

will call these function after writing the new vector, but

before returning to the client!

If there is no extra Forbid()/Disable(), you also must not

patch Supervisor() and Schedule()! (Like all private functions,

you shouldn't patch Schedule() anyhow!)

DEADLY has no effect.

REPORT Forbid() missing

SEVERITY Level 2

EXPLANATION You should generally forbid the task scheduling when you

are about to patch the system. There is a slight probab-

ility that your patched function is called by another task

while you are still building up the chain.

If you are patching functions which can be called by

interrupts, make sure to also Disable() the system for

the time that your patch is not completed.

BTW: don't forget to clear the instruction cache when you

have finished the chaining.

DEADLY has no effect.

1.63   PatchWork: exec.library / Vacate

e x e c . l i b r a r y Vacate

REPORT V39+ will be required

SEVERITY Level 0

EXPLANATION Vacate() is only working in V39 or higher. Don't use

it on older AmigaOS releases!

DEADLY has no effect.

1.64   PatchWork: gadtools.library

g a d t o o l s . l i b r a r y

_____

_____

These gadtools.library patches are implemented:

CreateContext

CreateGadgetA

GT_GetGadgetAttrsA

GT_RefreshWindow

GT_SetGadgetAttrsA

1.65   PatchWork: gadtools.library / CreateContext

g a d t o o l s . l i b r a r y CreateContext

_____

_____

REPORT glistpointer is not set to NULL

SEVERITY Level 2

EXPLANATION The glistpointer you pass in must be initialized to NULL.

DEADLY has no effect.

1.66   PatchWork: gadtools.library / CreateGadgetA

g a d t o o l s . l i b r a r y CreateGadgetA

_____

_____

REPORT illegal gadget kind 19

SEVERITY Level 2

EXPLANATION The gadget kind you've passed in, is unknown to GadTools

DEADLY has no effect.

_____

REPORT previous must not be NULL

SEVERITY Level 2

EXPLANATION You have to set the previous parameter!

DEADLY has no effect.

_____

REPORT newgad must not be NULL

SEVERITY Level 3

_____

EXPLANATION You have not passed in a NewGadget structure. This hit

will most probably also cause Enforcer hits.

DEADLY has no effect.

_____

REPORT no ng_VisualInfo given

SEVERITY Level 3

EXPLANATION You must pass a valid VisualInfo in your NewGadget

structure. This hit will most probably also cause

Enforcer hits.

DEADLY has no effect.

_____

REPORT no ng_TextAttr given

SEVERITY Level 3

EXPLANATION You must pass a valid TextAttr in your NewGadget

structure. This hit will most probably also cause

Enforcer hits.

DEADLY has no effect.

1.67   PatchWork: gadtools.library / GT_GetGadgetAttrsA

g a d t o o l s . l i b r a r y GT_GetGadgetAttrsA

_____
_____

REPORT set req to NULL for future compatibility

SEVERITY Level 2

EXPLANATION speaks for itself.

DEADLY has no effect.

1.68   PatchWork: gadtools.library / GT_RefreshWindow

g a d t o o l s . l i b r a r y GT_RefreshWindow

_____
_____

REPORT set req to NULL for future compatibility

SEVERITY Level 2

EXPLANATION speaks for itself.

DEADLY has no effect.

_____

1.69 PatchWork: gadtools.library / GT_SetGadgetAttrsA

g a d t o o l s . l i b r a r y GT_SetGadgetAttrsA

REPORT set req to NULL for future compatibility

SEVERITY Level 2

EXPLANATION speaks for itself.

DEADLY has no effect.

1.70 PatchWork: graphics.library

g r a p h i c s . l i b r a r y

These graphics.library patches are implemented:

AllocSpriteDataA

AreaEllipse also AreaCircle()

BestModeIDA

BitMapScale

ChangeSprite

ChangeVPBitMap

DrawEllipse also DrawCircle()

EraseRect

FreeColorMap

GetExtSpriteA

MakeVPort

RectFill

ScalerDiv

SetChipRev

SetFont

SetMaxPen

WaitBOVP

1.71 PatchWork: graphics.library / AllocSpriteDataA

g r a p h i c s . l i b r a r y AllocSpriteDataA

REPORT bitmap (h=10) isn't tall enough for sprite (h=16)

SEVERITY Level 3

EXPLANATION See AutoDocs! You can see the height of the BitMap and

the height of the sprite.

DEADLY has no effect.

1.72   PatchWork: graphics.library / AreaEllipse

g r a p h i c s . l i b r a r y AreaEllipse

——

——

REPORT ellipse radius must be > 0

SEVERITY Level 3

EXPLANATION Speaks for itself. Also see AutoDocs!

DEADLY has no effect.

REMEMBER that AreaCircle() is only a macro calling AreaEllipse()!

1.73   PatchWork: graphics.library / BestModeIDA

g r a p h i c s . l i b r a r y BestModeIDA

——

——

REPORT Tag BIDTAG_NominalWidth: 0 is not allowed

Tag BIDTAG_NominalHeight: 0 is not allowed

Tag BIDTAG_DesiredWidth: 0 is not allowed

Tag BIDTAG_DesiredHeight: 0 is not allowed

SEVERITY Level 3

EXPLANATION Will most probably cause a division by zero.

DEADLY has no effect.

——

REPORT Tag BIDTAG_NominalWidth: out of UWORD range

Tag BIDTAG_NominalHeight: out of UWORD range

Tag BIDTAG_DesiredWidth: out of UWORD range

Tag BIDTAG_DesiredHeight: out of UWORD range

SEVERITY Level 1

EXPLANATION These tags require an UWORD. The upper word will

most probably be ignored.

DEADLY has no effect.

1.74   PatchWork: graphics.library / BitMapScale

g r a p h i c s . l i b r a r y BitMapScale

_____

_____

REPORT bsa_XSrcFactor (=16395) must be within 1..16383

bsa_XDestFactor (=0) must be within 1..16383

bsa_YSrcFactor (=16777) must be within 1..16383

bsa_YDestFactor (=0) must be within 1..16383

SEVERITY Level 3

EXPLANATION No further comment required.

DEADLY has no effect.

_____

REPORT bsa_Flags is not 0

SEVERITY Level 3

EXPLANATION Don't ask, just do it!

DEADLY has no effect.

_____

REPORT Bug: cannot copy with width > 1024 (see autodocs)

SEVERITY Level 2

EXPLANATION BitMapScale is only capable to use the old Agnus'

blitter, so width is limited to 1024, if XSrcFactor

equals XDestFactor. You must accept this even if it

works fine under your graphics board emulation.

DEADLY has no effect.

_____

REPORT Bug: cannot expand in Y direction (see autodocs)

SEVERITY Level 2

EXPLANATION The Y range has been exceeded. See AutoDocs for

their calculation.

DEADLY has no effect.

1.75   PatchWork: graphics.library / ChangeSprite

g r a p h i c s . l i b r a r y ChangeSprite

_____

_____

REPORT spriteimage not initialized

SEVERITY Level 3

_____

EXPLANATION You must initialize the spriteimage structure as

described in the autodocs. Otherwise your sprite might

be messed up.

As only exception, this test is skipped if the sprite

has a height of 0 (i.e. it is hidden).

DEADLY has no effect.

REPORT spriteimage must be in CHIP RAM

SEVERITY Level 3

EXPLANATION ...since Amiga custom chips need it there. ;-)

DEADLY has no effect.

1.76   PatchWork: graphics.library / ChangeVPBitMap

g r a p h i c s . l i b r a r y ChangeVPBitMap

REPORT Current bitmap @0x12345678 and new bitmap @0x3456789A do not match

SEVERITY Level 3

EXPLANATION For double buffering, both bitmaps must match in

dimension, arrangement and depth. For better debugging,

you get a pointer to the currently displayed BitMap and

the BitMap structure that will replace it.

DEADLY has no effect.

1.77   PatchWork: graphics.library / DrawEllipse

g r a p h i c s . l i b r a r y DrawEllipse

REPORT ellipse radius must be > 0

SEVERITY Level 3

EXPLANATION Speaks for itself. Also see AutoDocs!

DEADLY has no effect.

REMEMBER that DrawCircle() is only a macro calling DrawEllipse()!

1.78 PatchWork: graphics.library / EraseRect

g r a p h i c s . l i b r a r y EraseRect

_____

_____

REPORT max must be >= min

SEVERITY Level 3

EXPLANATION xmax/ymax must not be lower than xmin/ymin!

DEADLY has no effect.

1.79 PatchWork: graphics.library / FreeColorMap

g r a p h i c s . l i b r a r y FreeColorMap

_____

_____

REPORT V39+ will be required

SEVERITY Level 0, MinOS 39

EXPLANATION if you pass a NULL pointer!

DEADLY has no effect.

1.80 PatchWork: graphics.library / GetExtSpriteA

g r a p h i c s . l i b r a r y GetExtSpriteA

_____

_____

REPORT V40+ will be required for proper operation

SEVERITY Level 0, MinOS 40

EXPLANATION This function is very buggy until V40.

DEADLY has no effect.

1.81 PatchWork: graphics.library / MakeVPort

g r a p h i c s . l i b r a r y MakeVPort

_____

_____

REPORT ViewPort->RasInfo has not been set

SEVERITY Level 3

EXPLANATION RasInfo must be valid!

DEADLY has no effect.

REPORT DUALPF ViewPort->RasInfo.Next has not been set

SEVERITY Level 3

EXPLANATION You must pass a second RasInfo if you use the Dual

Playfield mode.

DEADLY has no effect.

1.82 PatchWork: graphics.library / RectFill

g r a p h i c s . l i b r a r y RectFill

REPORT max must be >= min

SEVERITY Level 3

EXPLANATION xmax/ymax must not be lower than xmin/ymin!

DEADLY has no effect.

1.83 PatchWork: graphics.library / ScalerDiv

g r a p h i c s . l i b r a r y ScalerDiv

REPORT parameters are out of range

SEVERITY Level 3

EXPLANATION see AutoDocs.

DEADLY has no effect.

1.84 PatchWork: graphics.library / SetChipRev

g r a p h i c s . l i b r a r y SetChipRev

REPORT don't use this function!

SEVERITY Level 1

EXPLANATION Must be called only once. SetPatch does it for you.

DEADLY has no effect.

1.85   PatchWork: graphics.library / SetFont

g r a p h i c s . l i b r a r y SetFont

_____

REPORT NULL fonts are not allowed

SEVERITY Level 3

EXPLANATION Since V36, NULL fonts are not allowed any more. They

will produce Enforcer hits!

DEADLY has no effect.

_____

REPORT BTW: foobar.font/8 variant is obsoleted in V36+

SEVERITY Level 0

EXPLANATION Just a remember that this font does not meet the V36

requirements any more. SetFont will convert it for proper

use, but you shouldn't use this font any more.

DEADLY has no effect.

1.86   PatchWork: graphics.library / SetMaxPen

g r a p h i c s . l i b r a r y SetMaxPen

_____

REPORT maxpen==0 doesn't make much sense

SEVERITY Level 1

EXPLANATION No further comment.

DEADLY has no effect.

1.87   PatchWork: graphics.library / WaitBOVP

g r a p h i c s . l i b r a r y WaitBOVP

_____

REPORT busy wait, don't use if ever possible

SEVERITY Level 1

EXPLANATION In fact, this function busy waits until the desired

raster line has been reached. If you can avoid this

function, then you should do it by all means!

DEADLY has no effect.

REMARK This check has been temporarily disactivated, since

CyberGraphX does a clean emulation without busy loop.

1.88   PatchWork: intuition.library

i n t u i t i o n . l i b r a r y

_____

_____

These intuition.library patches are implemented:

CloseWindow

EasyRequestArgs

GadgetMouse

GetDefaultPubScreen

MakeClass

MakeScreen

ModifyIDCMP

RemakeDisplay

Request

RethinkDisplay

ScreenDepth

SetEditHook

SetMenuStrip

SetPointer

1.89   PatchWork: intuition.library / CloseWindow

i n t u i t i o n . l i b r a r y CloseWindow

_____

_____

REPORT still 4 Messages in the shared Window queue

SEVERITY Level 3

EXPLANATION This are two very serious mistakes in one. If you

use a shared Window MessagePort, you MUST reply all

pending messages, then clear the UserPort pointer and

set the IDCMP flags to 0. See the intuition AutoDocs

for a good example of how to safely close a shared

window.

It is most likely that this hit is followed by several

Enforcer and Mungwall hits.

DEADLY has no effect.

_____

REPORT MenuStrip @0x4567ABAC not cleared

SEVERITY Level 3

EXPLANATION You must clear the menu strip before closing the

window.

DEADLY has no effect.

_____

1.90   PatchWork: intuition.library / EasyRequestArgs

i n t u i t i o n . l i b r a r y EasyRequestArgs

——
——

REPORT no gadget specified
SEVERITY Level 3
EXPLANATION You must at least specify one gadget!
DEADLY has no effect.

——

REPORT es_StructSize is wrong (12 byte)
SEVERITY Level 2
EXPLANATION The es_StructSize field must be properly initialized.
Set the size of the EasyStruct.
DEADLY has no effect.

1.91   PatchWork: intuition.library / GadgetMouse

i n t u i t i o n . l i b r a r y GadgetMouse

——
——

REPORT obsoleted, improve your class implementation
SEVERITY Level 1
EXPLANATION This function is obsoleted, since the hook routines get
the mouse pointer position directly. It is recommended
not to call this function!
DEADLY has no effect.

1.92   PatchWork: intuition.library / GetDefaultPubScreen

i n t u i t i o n . l i b r a r y GetDefaultPubScreen

——
——

REPORT -
SEVERITY None, DEADLY mode only
EXPLANATION None
DEADLY This function usually returns a pointer to the public
screen, but you must not use this pointer for reading
the screen's structure! In DEADLY mode, NULL is returned
when GetDefaultPubScreen() returned NULL itself. Otherwise
0xFACEDEAD is returned, so you will get an Enforcer hit
when you try to read the screen structure.

——

1.93   PatchWork: intuition.library / MakeClass

i n t u i t i o n . l i b r a r y MakeClass

––

REPORT no superclass specified

SEVERITY Level 3

EXPLANATION You must specify the superclass of your class.

DEADLY has no effect.

1.94   PatchWork: intuition.library / MakeScreen

i n t u i t i o n . l i b r a r y MakeScreen

––

REPORT return code requires V39+

SEVERITY Level 0

EXPLANATION See autodocs.

DEADLY Always returns FALSE (non-zero).

1.95   PatchWork: intuition.library / ModifyIDCMP

i n t u i t i o n . l i b r a r y ModifyIDCMP

––

REPORT still 4 Messages in the shared Window queue

SEVERITY Level 3

EXPLANATION This is very serious. You MUST reply all pending messages

of the window BEFORE setting the IDCMP flags to 0.

It is most likely that this hit is followed by several

Enforcer and Mungwall hits.

Also see CloseWindow .

DEADLY has no effect.

REPORT return code requires V37+

SEVERITY Level 0

EXPLANATION See autodocs.

DEADLY has no effect, to guarantee an useable system.

1.96 PatchWork: intuition.library / RemakeDisplay

i n t u i t i o n . l i b r a r y RemakeDisplay

_____
_____

REPORT return code requires V39+

SEVERITY Level 0

EXPLANATION See autodocs.

DEADLY Always returns FALSE (non-zero).

1.97 PatchWork: intuition.library / Request

i n t u i t i o n . l i b r a r y Request

_____
_____

REPORT maximum of 8 requesters exceeded

SEVERITY Level 3

EXPLANATION Due to a bug in intuition, there is a maximum of 8 requesters

that are supported in a window and can be changed in

size, position, or depth.

DEADLY has no effect.

1.98 PatchWork: intuition.library / RethinkDisplay

i n t u i t i o n . l i b r a r y RethinkDisplay

_____
_____

REPORT return code requires V39+

SEVERITY Level 0

EXPLANATION See autodocs.

DEADLY Always returns FALSE (non-zero).

1.99 PatchWork: intuition.library / ScreenDepth

i n t u i t i o n . l i b r a r y ScreenDepth

_____
_____

REPORT reserved must be NULL

SEVERITY Level 3

EXPLANATION The parameter in A1 is reserved and must be set to NULL

for future compatibility!

DEADLY has no effect.

1.100   PatchWork: intuition.library / SetEditHook

i n t u i t i o n . l i b r a r y SetEditHook

_____

–––

REPORT risky function

SEVERITY Level 0

EXPLANATION See autodocs.

DEADLY has no effect.

1.101   PatchWork: intuition.library / SetMenuStrip

i n t u i t i o n . l i b r a r y SetMenuStrip

_____

–––

REPORT where is the menu?

SEVERITY Level 3

EXPLANATION You provided a NULL pointer as menu.

DEADLY has no effect.

1.102   PatchWork: intuition.library / SetPointer

i n t u i t i o n . l i b r a r y SetPointer

_____

–––

REPORT sprite data must be in CHIP ram

SEVERITY Level 3

EXPLANATION to be reachable by the custom chips. DraCo user

can ignore this hit.

DEADLY has no effect.

_____

REPORT width must be below 16

SEVERITY Level 3

EXPLANATION The hardware is limited to a width of 16 pixel.

DEADLY has no effect.

_____

REPORT hot spot is outside of the sprite

SEVERITY Level 0

EXPLANATION The provided offset defines a hot spot that lies outside

of the sprite borders and could irritate the user.

DEADLY has no effect.

_____

1.103   PatchWork: utility.library

u t i l i t y . l i b r a r y

_____

_____

These utility.library patches are implemented:

AllocNamedObjectA

MapTags

SDivMod32

UDivMod32

1.104   PatchWork: utility.library / AllocNamedObjectA

u t i l i t y . l i b r a r y AllocNamedObjectA

_____

_____

REPORT no object name specified

SEVERITY Level 3

EXPLANATION You must provide a string pointer in A0

DEADLY has no effect.

_____

REPORT bad ANO_Flags=0x12345678

SEVERITY Level 1

EXPLANATION You must only use NSF_NODUPS or NSF_CASE. All other bits

are reserved for future purposes.

DEADLY has no effect.

1.105   PatchWork: utility.library / MapTags

u t i l i t y . l i b r a r y MapTags

_____

_____

REPORT V39+ required for this mapType

SEVERITY Level 0

EXPLANATION Only MAP_KEEP_NOT_FOUND is functional until V39.

DEADLY has no effect.

1.106   PatchWork: utility.library / SDivMod32

u t i l i t y . l i b r a r y SDivMod32

_____

REPORT division by zero

SEVERITY Level 0

EXPLANATION See elementary school.

DEADLY has no effect.

1.107   PatchWork: utility.library / UDivMod32

u t i l i t y . l i b r a r y UDivMod32

_____

REPORT division by zero

SEVERITY Level 0

EXPLANATION See elementary school.

DEADLY has no effect.