Lasso - Support #48660

python3-lasso: segfault

18 novembre 2020 16:13 - Lorenzo Battistini

Statut:	Fermé	**Début:**	18 novembre 2020
Priorité:	Normal	**Echéance:**	
Assigné à:	Lorenzo Battistini	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Non	**Planning:**	Non

Description

Hello,

while loading my application, using **python3-lasso** version **2.5.1** on **ubuntu 18.04**, I get

```
[10676.087427] odoo[11449]: segfault at 18 ip 00000000004f7309 sp 00007f795d613130 error 4 in pyth
on3.6[400000+3b4000]
[10676.087435] Code: 3d 5b d1 57 00 00 0f 84 6e 02 00 00 48 8b 05 fe 78 57 00 48 8b 2d 07 d0 57 00
 48 89 c7 48 81 fd c0 17 63 00 0f 85 ef 02 00 00 <48> 8b 50 18 48 85 d2 0f 84 91 01 00 00 48 8b 5a
 30 48 85 db 0f 84
```

So I downloaded version **2.6.0**, compiled it, deployed **lasso.py** and **_lasso.so**, and:

```
# python3
Python 3.6.9 (default, Oct  8 2020, 12:12:24)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import lasso
Segmentation fault (core dumped)
```

Log file contains

```
[15184.943975] python3[23544]: segfault at 10 ip 00007fe2c002b2b9 sp 00007fffd6b62440 error 4 in l
ibapt-pkg.so.5.0.2[7fe2bff75000+1b6000]
[15184.943985] Code: 00 00 00 4c 89 70 10 4c 89 70 18 49 39 dd 74 49 90 bf 38 00 00 00 e8 96 c5 f8
 ff 48 89 c5 48 8d 78 10 48 8d 40 20 48 89 45 10 <48> 8b 73 10 48 8b 53 18 48 01 f2 e8 37 f5 ff ff
 8b 43 30 4c 89 f6
```

I also tried debugging with gdb:

```
Program received signal SIGSEGV, Segmentation fault.
0x00007ffff58722b9 in GlobalError::PushToStack() () from /usr/lib/x86_64-linux-gnu/libapt-pkg.so.5
.0
(gdb) backtrace
#0  0x00007ffff58722b9 in GlobalError::PushToStack() () at /usr/lib/x86_64-linux-gnu/libapt-pkg.so
.5.0
#1  0x00007ffff58f5fc2 in pkgInitConfig(Configuration&) () at /usr/lib/x86_64-linux-gnu/libapt-pkg
.so.5.0
#2  0x00007ffff639fed8 in  () at /usr/lib/python3/dist-packages/apt_pkg.cpython-36m-x86_64-linux-g
nu.so
#3  0x000000000050a12f in  ()
#4  0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#5  0x0000000000507be4 in  ()
#6  0x0000000000516069 in  ()
#7  0x0000000000566fae in PyCFunction_Call ()
#8  0x0000000000510e51 in _PyEval_EvalFrameDefault ()
#9  0x0000000000507be4 in  ()
#10 0x0000000000509900 in  ()
#11 0x000000000050a2fd in  ()
#12 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#13 0x00000000005095c8 in  ()
#14 0x000000000050a2fd in  ()
```

```
#15 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#16 0x00000000005095c8 in  ()
#17 0x000000000050a2fd in  ()
#18 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#19 0x00000000005095c8 in  ()
#20 0x000000000050a2fd in  ()
#21 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#22 0x0000000000508cd5 in _PyFunction_FastCallDict ()
#23 0x00000000005a4c61 in _PyObject_FastCallDict ()
#24 0x00000000005a5c9e in _PyObject_CallMethodIdObjArgs ()
#25 0x00000000004f6d6d in PyImport_ImportModuleLevelObject ()
#26 0x000000000050ddbf in _PyEval_EvalFrameDefault ()
#27 0x0000000000507be4 in  ()
#28 0x0000000000516069 in  ()
#29 0x0000000000566fae in PyCFunction_Call ()
#30 0x0000000000510e51 in _PyEval_EvalFrameDefault ()
#31 0x0000000000507be4 in  ()
#32 0x0000000000509900 in  ()
#33 0x000000000050a2fd in  ()
#34 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#35 0x00000000005095c8 in  ()
#36 0x000000000050a2fd in  ()
#37 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#38 0x00000000005095c8 in  ()
#39 0x000000000050a2fd in  ()
#40 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#41 0x00000000005095c8 in  ()
#42 0x000000000050a2fd in  ()
#43 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#44 0x0000000000508cd5 in _PyFunction_FastCallDict ()
#45 0x00000000005a4c61 in _PyObject_FastCallDict ()
#46 0x00000000005a5c9e in _PyObject_CallMethodIdObjArgs ()
#47 0x00000000004f6d6d in PyImport_ImportModuleLevelObject ()
#48 0x000000000050ddbf in _PyEval_EvalFrameDefault ()
#49 0x0000000000507be4 in  ()
#50 0x0000000000516069 in  ()
#51 0x0000000000566fae in PyCFunction_Call ()
#52 0x0000000000510e51 in _PyEval_EvalFrameDefault ()
#53 0x0000000000507be4 in  ()
#54 0x0000000000509900 in  ()
#55 0x000000000050a2fd in  ()
#56 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#57 0x00000000005095c8 in  ()
#58 0x000000000050a2fd in  ()
#59 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#60 0x00000000005095c8 in  ()
#61 0x000000000050a2fd in  ()
#62 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#63 0x00000000005095c8 in  ()
#64 0x000000000050a2fd in  ()
#65 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#66 0x0000000000508cd5 in _PyFunction_FastCallDict ()
#67 0x00000000005a4c61 in _PyObject_FastCallDict ()
#68 0x00000000005a5c9e in _PyObject_CallMethodIdObjArgs ()
#69 0x00000000004f6d6d in PyImport_ImportModuleLevelObject ()
#70 0x000000000050ddbf in _PyEval_EvalFrameDefault ()
#71 0x0000000000507be4 in  ()
#72 0x0000000000516069 in  ()
#73 0x0000000000566fae in PyCFunction_Call ()
#74 0x0000000000510e51 in _PyEval_EvalFrameDefault ()
#75 0x0000000000507be4 in  ()
#76 0x0000000000509900 in  ()
#77 0x000000000050a2fd in  ()
#78 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#79 0x00000000005095c8 in  ()
#80 0x000000000050a2fd in  ()
#81 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
```

```
#82 0x00000000005095c8 in  ()
#83 0x000000000050a2fd in  ()
#84 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#85 0x00000000005095c8 in  ()
#86 0x000000000050a2fd in  ()
#87 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#88 0x0000000000508cd5 in _PyFunction_FastCallDict ()
#89 0x00000000005a4c61 in _PyObject_FastCallDict ()
#90 0x00000000005a5c9e in _PyObject_CallMethodIdObjArgs ()
#91 0x00000000004f6d6d in PyImport_ImportModuleLevelObject ()
#92 0x000000000050ddbf in _PyEval_EvalFrameDefault ()
#93 0x0000000000507be4 in  ()
#94 0x0000000000516069 in  ()
#95 0x0000000000566fae in PyCFunction_Call ()
#96 0x0000000000510e51 in _PyEval_EvalFrameDefault ()
#97 0x0000000000507be4 in  ()
#98 0x0000000000509900 in  ()
#99 0x000000000050a2fd in  ()
#100 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#101 0x00000000005095c8 in  ()
#102 0x000000000050a2fd in  ()
#103 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#104 0x00000000005095c8 in  ()
#105 0x000000000050a2fd in  ()
#106 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#107 0x00000000005095c8 in  ()
#108 0x000000000050a2fd in  ()
#109 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#110 0x0000000000508cd5 in _PyFunction_FastCallDict ()
#111 0x00000000005a4c61 in _PyObject_FastCallDict ()
#112 0x00000000005a5c9e in _PyObject_CallMethodIdObjArgs ()
#113 0x00000000004f6d6d in PyImport_ImportModuleLevelObject ()
#114 0x00000000005140a4 in  ()
#115 0x0000000000566f73 in PyCFunction_Call ()
#116 0x0000000000510e51 in _PyEval_EvalFrameDefault ()
#117 0x0000000000507be4 in  ()
#118 0x0000000000509900 in  ()
#119 0x000000000050a2fd in  ()
#120 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#121 0x00000000005095c8 in  ()
#122 0x000000000050a2fd in  ()
#123 0x000000000050beb4 in _PyEval_EvalFrameDefault ()
#124 0x0000000000508cd5 in _PyFunction_FastCallDict ()
#125 0x00000000005a4c61 in _PyObject_FastCallDict ()
#126 0x00000000005a5c9e in _PyObject_CallMethodIdObjArgs ()
#127 0x00000000004f6d6d in PyImport_ImportModuleLevelObject ()
#128 0x000000000050ddbf in _PyEval_EvalFrameDefault ()
#129 0x0000000000508cd5 in _PyFunction_FastCallDict ()
#130 0x00000000005a4c61 in _PyObject_FastCallDict ()
#131 0x000000000063831b in PyErr_PrintEx ()
#132 0x0000000000638703 in PyRun_SimpleFileExFlags ()
#133 0x0000000000639281 in Py_Main ()
#134 0x00000000004b0dc0 in main ()
```

But I don't know how to proceed.

Any suggestion?

Thanks

Historique

#3 - 18 novembre 2020 17:40 - Frédéric Péters

[15184.943975] python3[23544]: segfault at 10 ip 00007fe2c002b2b9 sp 00007fffd6b62440 error 4 in libapt-pkg.so.5.0.2[7fe2bff75000+1b6000]

It says it crashes in libapt-pkg, I don't know how that came there, some kind of automatic segv handler?

Could you install debug packages, to get symbols on all backtrace lines?

I guess you won't get an answer if you report this against the ubuntu package but if you are able to run this with Debian it would be helpful and you could then use the debian bug tracking system to report the problem; also debian buster lasso package is already on 2.6.0 (and there's even an official stretch backport).

#4 - 18 novembre 2020 18:20 - Benjamin Dauvergne

There are also some rewrite to the python binding in the master branch that you could try to see if it fixes some of your problems, https://git.entrouvert.org/lasso.git/log/ . These changes are not released yet.

#5 - 19 novembre 2020 07:01 - Lorenzo Battistini

> Could you install debug packages, to get symbols on all backtrace lines?

How can I install debug packages?

> if you are able to run this with Debian it would be helpful and you could then use the debian bug tracking system to report the problem; also debian buster lasso package is already on 2.6.0 (and there's even an official stretch backport).

At the moment our servers are on Ubuntu 18.04 so we would need to make it work there.

> There are also some rewrite to the python binding in the master branch that you could try to see if it fixes some of your problems, https://git.entrouvert.org/lasso.git/log/ .

Ok, downloaded. **configure** is not present, so I run **./autogen.sh**:

```
* Running libtoolize
libtoolize: putting auxiliary files in '.'.
libtoolize: copying file './ltmain.sh'
libtoolize: putting macros in AC_CONFIG_MACRO_DIRS, 'm4'.
libtoolize: copying file 'm4/libtool.m4'
libtoolize: copying file 'm4/ltoptions.m4'
libtoolize: copying file 'm4/ltsugar.m4'
libtoolize: copying file 'm4/ltversion.m4'
libtoolize: copying file 'm4/lt~obsolete.m4'
You don't have gtk-doc installed, and thus
won't be able to generate the documentation.
* Running aclocal-1.15
* Running autoconf
* Running automake-1.15
configure.ac:93: installing './compile'
configure.ac:23: installing './config.guess'
configure.ac:23: installing './config.sub'
configure.ac:31: installing './install-sh'
configure.ac:31: installing './missing'
bindings/java/Makefile.am: installing './depcomp'
parallel-tests: installing './test-driver'
docs/reference/lasso/Makefile.am:98: error: EXTRA_DIST must be set with '=' before using '+='
```

So, **./configure** ends with

```
config.status: error: cannot find input file: `docs/reference/lasso/Makefile.in'
```

What is the correct procedure to compile from master?

Thanks!

#6 - 19 novembre 2020 08:32 - Frédéric Péters

> How can I install debug packages?

I don't use Ubuntu but https://wiki.ubuntu.com/Debug%20Symbol%20Packages

> What is the correct procedure to compile from master?

I wouldn't mix packages and a built-from-source version; proper procedure here would be to create a new package based on a more recent lasso release.

However I just got access to an ubuntu 18.04 server and checked and it loaded properly:

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libegl-mesa0 libegl1 libgbm1 libgeos-3.6.2 libllvm7 libwayland-client0 libwayland-egl1 libwayland-egl1-mesa
libwayland-server0
  libxcb-xfixes0
Use 'apt autoremove' to remove them.
The following additional packages will be installed:
  liblasso3
The following NEW packages will be installed:
  liblasso3 python3-lasso
0 upgraded, 2 newly installed, 0 to remove and 103 not upgraded.
Need to get 277 kB of archives.
After this operation, 1623 kB of additional disk space will be used.
Do you want to continue? [Y/n]
Get:1 http://ubuntu.mirrors.ovh.net/ubuntu bionic-updates/main amd64 liblasso3 amd64 2.5.1-0ubuntu1.1 [160 kB]
Get:2 http://ubuntu.mirrors.ovh.net/ubuntu bionic-updates/universe amd64 python3-lasso amd64 2.5.1-0ubuntu1.1
[116 kB]
Fetched 277 kB in 0s (945 kB/s)
Selecting previously unselected package liblasso3.
(Reading database ... 98428 files and directories currently installed.)
Preparing to unpack .../liblasso3_2.5.1-0ubuntu1.1_amd64.deb ...
Unpacking liblasso3 (2.5.1-0ubuntu1.1) ...
Selecting previously unselected package python3-lasso.
Preparing to unpack .../python3-lasso_2.5.1-0ubuntu1.1_amd64.deb ...
Unpacking python3-lasso (2.5.1-0ubuntu1.1) ...
Setting up liblasso3 (2.5.1-0ubuntu1.1) ...
Setting up python3-lasso (2.5.1-0ubuntu1.1) ...
Processing triggers for libc-bin (2.27-3ubuntu1.2) ...
# python3
Python 3.6.9 (default, Oct  8 2020, 12:12:24)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import lasso
>>>
```

Probably there's something particular on your current system; do you reproduce on a fresh installation of ubuntu 18.04?

#7 - 19 novembre 2020 10:47 - Lorenzo Battistini

> However I just got access to an ubuntu 18.04 server and checked and it loaded properly

I can install **python3-lasso** version **2.5.1** and execute **import lasso** without problem too, but I get segfault while loading my application that uses **python3-lasso** (This is the original problem).

I debugged my application and found that segfault is raised at this line
https://github.com/python/cpython/blob/dae5d728bc3f1d4039b64e4ec3a9036fd5d19587/Lib/posixpath.py#L96
while returning a path not directly related to python3-lasso.

Other segfault are raised while performing other operations in other parts of the application, not related to python3-lasso; maybe something related reading files from file system.
The fact is: if I remove python3-lasso, segmentation faults disappear.

So I really can't understand why this is happening.

Thus, I tried to update python3-lasso to 2.6.0 but without success.

How do you suggest to proceed?

Thanks!

#8 - 19 novembre 2020 10:57 - Benjamin Dauvergne

Lorenzo Battistini a écrit :

> Thus, I tried to update python3-lasso to 2.6.0 but without success.

Would you be able to produce a minimal python example which reproduce the segfault on your side ? If we could reproduce it on 2.5.1 as you do we could start looking for a solution (or even see if things are effectively fixed on master).

#9 - 19 novembre 2020 11:47 - Lorenzo Battistini

> Would you be able to produce a minimal python example which reproduce the segfault on your side ? If we could reproduce it on 2.5.1 as you do we could start looking for a solution (or even see if things are effectively fixed on master).

Ok, I reproduced it:

```
Python 3.6.9 (default, Oct  8 2020, 12:12:24)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from lxml import etree
>>> etree.parse('import_xml.rng')
<lxml.etree._ElementTree object at 0x7f23f4347088>
>>> import lasso
>>> etree.parse('import_xml.rng')
Segmentation fault (core dumped)
```

with **lxml==3.7.1**
and **import_xml.rng** is this file
https://github.com/odoo/odoo/blob/64ff41ce805c72eb700b45d55adfc7379de56862/odoo/import_xml.rng

Thanks

#10 - 17 décembre 2020 14:26 - Benjamin Dauvergne

- Statut changé de Nouveau à Information nécessaire

- Assigné à mis à Lorenzo Battistini

- Priorité changé de Haut à Normal

I'm sorry but I'm not able to reproduce with 2.5.1, 2.6.0 or 2.6.1 on Debian Buster, you should open an issue on Ubuntu or try to reproduce it from upstream source code with --enable-debugging to have a full trace.

#11 - 18 décembre 2020 11:29 - Lorenzo Battistini

Hello,
we replaced **python3-lasso** with **pysaml2** so I will not work on this.
Thank you anyway

#12 - 18 décembre 2020 11:34 - Benjamin Dauvergne

- Statut changé de Information nécessaire à Fermé