After an evaluation, GNOME has moved from Bugzilla to GitLab. Learn more about GitLab.

No new issues can be reported in GNOME Bugzilla anymore.
To report an issue in a GNOME project, go to GNOME GitLab.

Do **not** go to GNOME Gitlab for: Bluefish, Doxygen, GnuCash, GStreamer, java-gnome, LDTP, NetworkManager, Tomboy.

Bug 772726 - (CVE-2016-9318) XXE problems continue

Status: RESOLVED FIXED

Product: libxml2
Component: general
Version: git master
Hardware: Other All

Importance: High critical
Target Milestone: ---
Assigned To: Daniel Veillard
QA Contact: libxml QA maintainers

URL:
Whiteboard: CVE-2016-9318

Depends on:
Blocks:

Reported: 2016-10-11 04:38 UTC by Aleksey Sanin
Modified: 2019-04-23 11:02 UTC

See Also: https://github.com/sparklemotion/nokogiri/issues/1582

GNOME target: ---
GNOME version: ---

Attachments

proposal to fix ctxt->external and support XML_PARSE_NOXXE (7.32 KB, patch) 2017-04-26 06:02 UTC, dmoppert	none	Details	Review
xxe test script (2.45 KB, application/x-compressed-tar) 2017-06-15 03:56 UTC, dmoppert		Details	
(partial) fix for the XXE (389 bytes, patch) 2017-12-05 05:31 UTC, Aleksey Sanin	none	Details	Review

Aleksey Sanin 2016-10-11 04:38:21 UTC Description

Started as xmlsec bug https://github.com/lsh123/xmlsec/issues/43

The external entity is loaded from here:

(gdb) where

+ Trace 236734

There is a check in xmlParserHandlePEReference() that tries to handle this condition:

```
                    if ((entity->etype == XML_EXTERNAL_PARAMETER_ENTITY) &&
                        ((ctxt->options & XML_PARSE_NOENT) == 0) &&
                        ((ctxt->options & XML_PARSE_DTDVALID) == 0) &&
                        ((ctxt->options & XML_PARSE_DTDLOAD) == 0) &&
                        ((ctxt->options & XML_PARSE_DTDATTR) == 0) &&
                        (ctxt->replaceEntities == 0) &&
                        (ctxt->validate == 0))
                        return;
```

But I think it is too restrictive and it should be changed to something simpler like

https://git.gnome.org/browse/libxml2/commit/?id=b1d34de46a11323fccffa9fadeb33be670d602f5
https://git.gnome.org/browse/libxml2/commit/?id=4629ee02ac649c27f9c0cf98ba017c6b5526070f

Aleksey Sanin 2016-10-11 05:14:30 UTC Comment 1

The following patch fixes the issue, however there might be other similar problems worth an audit. \

```
diff --git a/parserInternals.c b/parserInternals.c
index bfc778a..bade1f3 100644
--- a/parserInternals.c
+++ b/parserInternals.c
@@ -1438,6 +1438,11 @@ xmlNewEntityInputStream(xmlParserCtxtPtr ctxt, xmlEntityPtr
entity) {
                break;
             case XML_EXTERNAL_GENERAL_PARSED_ENTITY:
             case XML_EXTERNAL_PARAMETER_ENTITY:
+                if (((ctxt->options & XML_PARSE_NOENT) == 0) &&
+                    ((ctxt->options & XML_PARSE_DTDVALID) == 0)) {
+                    xmlErrInternal(ctxt, "xmlNewEntityInputStream will not read
content for external entity\n",
+                                   NULL);
+                }
                return(xmlLoadExternalEntity((char *) entity->URI,
                       (char *) entity->ExternalID, ctxt));
             case XML_INTERNAL_GENERAL_ENTITY:
```

Daniel Veillard 2016-10-11 06:57:25 UTC Comment 2

Hum, the patch is just trying to address the problem at another layer.
But I don't understand why the existing test is not catching it,
it seems to imply that within xmlsec1 context we have either
XML_PARSE_DTDLOAD or XML_PARSE_DTDATTR or ctxt->replaceEntities
or ctxt->validate. All of them asks for loading of DTDs and then
fetching the external subset.
 I wonder if we aren't activating XML_PARSE_DTDATTR to have the
defaulted attributes from the DTD included in the canonical form,
in which case I don't see how we could both prevent fetching external
subset and be compliant with the spec. I think it's worth investigating
because that mean the patch leads to a regression from an XML Sec POV.

  Could you clarify ?

    thanks,

Daniel

Aleksey Sanin 2016-10-11 17:07:23 UTC Comment 3

The problem is also reproducible with xmllint. Thinking more about it, I think it is a valid scenario where one might want to parse the document, validate it, etc. but do not allow the parser to read any local or network files. I think the current patches (see the links in my first comment) are trying to guess the case when local/network access should be disabled based on other flags. I think it might be better to have an explicit (default?) flag that limits parser to only process the current file/memory block. What do you think?

Matias Brutti 2016-10-11 18:22:11 UTC Comment 4

Daniel & Aleksey, I am the author of the bug reported to xmlsec1. I think the best path here is to make it secure by default and have an optional flag or method that allows the developer enable those whenever necessary, but the default behavior should be NOT to parse them.

Matias Brutti 2016-10-31 19:26:15 UTC Comment 5

Has there been any update / decision on this bug ? Also, what is the process for requesting a CVE for this vulnerability ?

Matthew Valdes 2016-11-10 19:51:44 UTC Comment 6

Matias, please request a CVE through https://cve.mitre.org/cve/request_id.html

According to the following, it appears that XXE was thought to be disabled by default.

https://mail.gnome.org/archives/xml/2012-October/msg00045.html
http://git.gnome.org/browse/libxml2/commit/?id=4629ee02ac649c27f9c0cf98ba017c6b5526070f.

Matias Brutti 2016-11-11 17:31:17 UTC Comment 7

Matthew, I requested it yesterday. CVE Request 261315. I will update the ticket with a proper CVE ID once I have one.

Matias Brutti 2016-11-15 16:47:29 UTC Comment 8

The CVE ID for this vuln is CVE-2016-9318

dmoppert 2016-11-28 02:41:40 UTC Comment 9

There seem to be some misunderstandings around this issue. The way I understand the upstream xmlsec ticket, the problem here is external entity expansion specifically with respect to parameter entities - similar to CVE-2014-0191. I believe the code referenced in comment #0 was originally introduced to address this CVE.

To reproduce the problem described @ xmlsec with xmllint, I need to use the --postvalid option. With --postvalid, all the original CVE-2014-0191 reproducers are triggered, indicating that DTD validation does not correctly respect XML_PARSE_NOENT.

The patch in comment 1 does NOT change this behaviour.

I think the identified block in parser.c:xmlParserHandlePEReference() should begin:

```
    // refuse to expand any entities if NOENT is not provided
    // this is the same behaviour as expected for normal entities
    if ((ctxt->options & XML_PARSE_NOENT) == 0))
        return;
```

With this change:
- `xmllint --postvalid` no longer loads external parameter entities
- `xmllint --noent --nonet --postvalid` no longer loads external parameter entities from HTTP URLs, but it does load them from the filesystem

I don't really understand the rest of the flags, but I'm tempted to guess they should be joined with || instead of && in the test that's currently there. That seems to make sense from my limited knowledge, except DTDATTR - I have no idea what that is doing here.

```
                    if ((entity->etype == XML_EXTERNAL_PARAMETER_ENTITY) && (
                          ((ctxt->options & XML_PARSE_NOENT) == 0) ||
                          ((ctxt->options & XML_PARSE_DTDVALID) == 0) ||
                          ((ctxt->options & XML_PARSE_DTDLOAD) == 0) ||
                          ((ctxt->options & XML_PARSE_DTDATTR) == 0) ||
                          (ctxt->replaceEntities == 0) ||
                          (ctxt->validate == 0)))
                        return;
```

I agree that a flag similar to NONET which disables fetching *all* external resources is a good idea. This could be handled simply in xmlIO.c, prior to the XML_PARSE_NONET test. That probably belongs on another ticket as RFE - getting NOENT honoured for parameter entities would be a good outcome for this one.

dmoppert 2016-11-28 03:00:23 UTC Comment 10

To summarise my previous comment briefly, I think there are two bugs at play here:

#1: XML_PARSE_NOENT=0 is not honoured for parameter entities when validating DTD

#2: no means are provided to prevent expanding file-based entities, as XML_PARSE_NOENT=1 does for network-based entities

Daniel Veillard 2016-11-28 10:52:30 UTC Comment 11

--postvalid of xmllint requires loading the DTD and hence is incompatible with --noent and equivalent, that's a fact of XML validation definition.

Think about it, you ask to validate against the DTD after parsing but you ask to not load the DTD, entities are defined in the DTD.

To validate with the DTD you *MUST* load entities, so 1 is obvious at least if understanding XML.
XML_PARSE_NOENT is about all entities, local, remote alike.

W.r.t. patch
"`xmllint --postvalid` no longer loads external parameter entities"
then that's broken. If you ask to validate you must load the entities parameter or general ones.

Daniel

dmoppert 2016-11-29 01:07:50 UTC Comment 12

> To validate with the DTD you *MUST* load entities

You're absolutely right - my point #1 didn't make any sense. Sorry for the noise.

In reference to #2 (and comment 3), is there a recommended way for clients to limit or disallow what external files can be opened? A flag to altogether disable file-based entities might be too blunt for tools that need (for instance) to validate

against a local store of DTDs. If there's a way to control this in the existing API, xmlsec and other tools could surely benefit from it.

Aleksey Sanin 2016-11-29 02:27:40 UTC Comment 13

> To validate with the DTD you *MUST* load entities

Correct, however for security reasons I might want to ONLY allow embedded references. I.e. for an XML file received from the user I don't want it to try to read files on local filesystem or remotely over the network. Of course, if external entities are present (and loading of remote entities is disabled) then this should result in an error.

dmoppert 2016-12-01 07:03:41 UTC Comment 14

I have been looking for ways within the existing API to support the mode suggested in this ticket: parsing a document (and optionally validating internal DTD) with entities being expanded, but forbidding the loading of any external entities, whether file-based or network-based. My prototype adds a flag XML_PARSE_NOXXE, exposed via `xmllint --noxxe` for basic testing.

xmlSetExternalEntityLoader() looks like the right choice: this is how XML_PARSE_NONET does its magic, so providing a similar function xmlNoXxeExternalEntityLoader() seems the right way to limit what external URIs libxml2 will open. But I'm running into some difficulties; I hope you can provide some advice on how best to proceed (or if there's an altogether better approach).

The trouble I'm encountering is that xmlLoadExternalEntity() is used within xmlCreateURLParserCtxt(), setting up the parser for the initial document. I want to allow this call, but forbid (by raising an error a la xmlNoNetExternalEntityHandler) calls that are actually for external entities.

Examining the ctxt structure in breakpoints, I can't see a field that confidently provides this information. ctxt->external sounds right, but this is undermined by xmlParseExternalEntityPrivate() doing `ctxt->external = oldctxt->external`.

Questions:

#1 - does my approach seem sound?

#2 - is ctxt->external intended to give the information I want? Patching xmlParseExternalEntityPrivate() doesn't seem to immediately cause any regressions, but I'm not sure this is right - and if it is, whether it should be set to 1 or 2.

#3 - if not ctxt->external, is there something else in ctxt that can be checked? input_id is tempting, but I don't know if contexts can be re-used.

#4 - how should ExternalEntityHandler functions deal with ctxt==NULL? I started by returning NULL after the pattern established in other functions, but ran into some errors in the Python test suite.

If I'm on the right track and can get some guidance on the above questions, I should be able to attach a working patch in the next few days and start working on proper test coverage.

Thanks!

dmoppert 2016-12-13 01:56:19 UTC Comment 15

Created ~~attachment 341856 [details]~~ [review]
proposal for XML_PARSE_NOXXE

Attached a proposed patch that uses (ctxt->input_id != 1) to determine whether an external reference is being resolved. I'm not convinced this is the right approach, but (ctxt->external) did not give the results I expected.

Internally, this adds a flag XML_PARSE_NOXXE, exposed via xmllint as "-noxxe". No regressions in the existing test suite, and passes my ad-hoc tests for XXE using xmllint.

Note that the patch doesn't depend on any libxml2 internals, so the same approach could be used by downstream projects like xmlsec without waiting on a new release of libxml2. Assuming it's correct, or can be made so.

Gengxiang Meng 2016-12-27 05:18:41 UTC Comment 16

I check the solution and have one confusion, the change in libxml/libxml2-2.9.4/xmlIO.c:

```
+        if (options & XML_PARSE_NOXXE) {
+            ctxt->options -= XML_PARSE_NOXXE;
+            ret = xmlNoNetExternalEntityLoader(URL, ID, ctxt);
+            ctxt->options = options;
+            return(ret);
+        }
+
```

should be change to:

```
+        if (options & XML_PARSE_NOXXE) {
```

```
+            ctxt->options -= XML_PARSE_NOXXE;
+            ret = xmlNoXxeExternalEntityLoader(URL, ID, ctxt);
+            ctxt->options = options;
+            return(ret);
+        }
+
```

Not sure, let me know if I am wrong, many thanks.

dmoppert 2016-12-28 06:58:28 UTC Comment 17

Created ~~attachment 342521~~ ~~[details]~~ [review]
proposal for XML_PARSE_NOXXE (updated)

Gengxian Meng,

You are right - thanks for picking that up. The XML_PARSE_NOXXE block should be calling xmlNoXxeExternalEntityLoader, not xmlNoNet...

Updated patch attached.

Daniel Veillard 2017-04-07 14:57:16 UTC Comment 18

Okay, reviewed the patch ... finally.
It's always a risk to add one more parsing option but in that case fine.
I checked the patch, changed a couple of cosmetic things extra tag and comments as this is not entitie substittion we are forbidding but more drastically entity loading.

Commited as
https://git.gnome.org/browse/libxml2/commit/?id=2304078555896cf1638c628f50326aeef6f0e0d0

thanks !

Daniel

storysoft 2017-04-13 00:49:37 UTC Comment 19

Did you reproduce the error?

Nick Wellnhofer 2017-04-20 14:31:15 UTC Comment 20

What about this unrelated change to xmlNoNetExternalEntityLoader?

```
@@ -4160,6 +4170,13 @@ xmlNoNetExternalEntityLoader(const char *URL, const char
*ID,
     xmlParserInputPtr input = NULL;
     xmlChar *resource = NULL;

+    if (ctxt == NULL) {
+        return(NULL);
+    }
+    if (ctxt->input_id == 1) {
+        return xmlDefaultExternalEntityLoader((const char *) URL, ID, ctxt);
+    }
+
 #ifdef LIBXML_CATALOG_ENABLED
     resource = xmlResolveResourceFromCatalog(URL, ID, ctxt);
 #endif
```

It seems to me that this allows to load the initial document over the network even if XML_PARSE_NONET was supplied. Also, xmlNoXxeExternalEntityLoader seems to allow loading a document over network if both XML_PARSE_NONET and XML_PARSE_NOXXE were supplied.

dmoppert 2017-04-26 05:58:14 UTC Comment 21

Argh. Having put it through a thorough test suite, it looks like this patch is a mess. It doesn't work as advertised, and causes a regression:

I don't get the (In reply to Nick Wellnhofer from comment #20)
> What about this unrelated change to xmlNoNetExternalEntityLoader?

That should not have been present - you are right, it's a change in behaviour as you describe:

> allows to load the initial document over the network even if XML_PARSE_NONET was supplied.

> Also, xmlNoXxeExternalEntityLoader seems to allow loading a document over
> network if both XML_PARSE_NONET and XML_PARSE_NOXXE were supplied.

I don't understand this claim? In my testing that is not the case; the relevant code is in xmlDefaultExternalEntityLoader which checks NOXXE before NONET. I believe this is correct (specifying both, the more restrictive NOXXE will take precedence).

Updated patch and test script coming ..

dmoppert 2017-04-26 06:02:14 UTC Comment 22

Created attachment 350433 [details] [review]
proposal to fix ctxt->external and support XML_PARSE_NOXXE

This patch is similar to ~~attachment 342521 [details]~~ [review], but it uses ctxt->external as that appears to be what it is designed for (input_nr is not reliable for this purpose).

To use ctxt->external, this flag needs to be set correctly in xmlCreateEntityParserCtxtInternal and not copied from the parent context in xmlParseExternalEntityPrivate.

Unrelated changes to xmlNoNetExternalEntityLoader have been removed.

dmoppert 2017-04-26 06:11:38 UTC Comment 23

I'd like to add the test cases + runner as a private attachment, at least until a proper patch is merged. In the meantime, libxml2 developers can contact me directly.

Test cases consist of 6 xml files run through xmllint with 16 combinations of flags, expecting particular outcomes for each run (external entity loaded or not loaded). With attachment 350433 [details] [review], behaviour for existing flags is consistent with previous builds and the new `--noxxe` flag stops XXE in all cases.

Advice on how to integrate such tests into the libxml2 `make check` test suite would also be appreciated - that's where they ultimately belong, but a Tcl script probably isn't the ideal way to package them.

David Kilzer 2017-05-30 23:04:19 UTC Comment 24

Reopening this bug since Attachment 350433 [details] needs review and landing. Also marking the bug as security once again.

To add the tests to `make check`, you'll need to update runtest.c (or one of the other 5 test tools: runxmlconf, testapi, testchar, testdict, testrecurse) to add the test mode. I suspect runtest.c will be the closest match.

I did a similar thing in ~~Bug 776895~~ (patch to be posted soon) where I added `make Schemastests` for `make tests`, and added support to runtest.c for "Schemas streaming regression tests" that did the same thing.

I'm actually not sure what the future direction of `make check` vs. `make tests` is, so I added it to both.

What I really want is a test mode where I can run the test case through xmllint, but specify the command-line switches to use in the test (or in a metadata file in the test directory). Then we don't have to add all these single-mode test directories (even though some are very useful).

Nick Wellnhofer 2017-06-05 21:41:55 UTC Comment 25

Maybe it makes more sense to tackle this at the parser level, not at the IO level. If XML_PARSE_NOXXE is set, the parser could simply refuse to call xmlLoadExternalEntity.

Daniel Veillard 2017-06-08 20:31:51 UTC Comment 26

That might be the simplest indeed !

Daniel

dmoppert 2017-06-15 03:56:20 UTC Comment 27

Created attachment 353798 [details]
xxe test script

dmoppert 2017-06-15 04:17:23 UTC Comment 28

(ad-hoc, not integrated) Test script attached above. Note that my most recent attempt at a patch still fails - I inverted the logic around using ctxt->external, causing xmllint to reject opening the initial file. I expanded the test to also serve the input through a FIFO so each test can ensure that is read.

One problem I have struck is that xmlLoadExternalEntity() is called by xmlCreateURLParserCtxt() from xmlReadFile() to load the initial document. This flailing about with ctxt fields has been trying to find a way to identify when xLEE() is being called for an actual external entity or not, but so far my attempts have gone awry.

Nick Wellnhofer 2017-06-15 11:14:24 UTC Comment 29

> This flailing about with ctxt fields has been trying to find a way to identify when xLEE() is being called for an actual external entity

I simply don't see a way to do that without additional changes. ctxt->input_id doesn't work because some external entites are loaded with a fresh parser context, others are not. The fresh parser context is indistinguishable from the context used to load the initial document.

Aleksey Sanin 2017-12-05 05:31:14 UTC Comment 30

Created attachment 364995 [details] [review]
(partial) fix for the XXE

This solves at least part of the problems.

Nikita Nirbhavane 2017-12-05 06:12:34 UTC Comment 31

Hi Aleksey,

That's great. This solves partial problems. When can we expect a complete fix for the vulnerability?

Nick Wellnhofer 2017-12-05 10:48:17 UTC Comment 32

This isn't really a vulnerability. libxml2 only loads external entities if the user requests to substitute entities or to validate a document. In this case libxml2 has always given the user full control over which external entities are loaded. This bug is only about making it easier to completely suppress loading of external entities.

Daniel Veillard 2017-12-08 08:47:34 UTC Comment 33

Okay, I checked through the code where one would create sub-parsers of
existing parser or parsing outside the context of a normal parsing
framework. I found a few other places where input_id ought to be increased
or set higher than 1. Note that this wasn't used for entity snowballing
detection but to make sure things like opening/closing constructs where
happening in the same entity. That said not it should be usable in
a reliable way for your purpose (Famous Last Words <grin/>)

Commited as
https://git.gnome.org/browse/libxml2/commit/?id=ad88b54f1a28a8565964a370b5d387927b633c0d

Aleksey, please give it a try, then I may try to push this month release !

thanks,

Daniel

Aleksey Sanin 2017-12-08 15:33:31 UTC Comment 34

I just tried and all the test cases I know about are covered. Do you know about any other places where we create parsing context?

Nikunj Mavani 2018-05-22 08:24:52 UTC Comment 35

As we are waiting for the fix of this CVE from quite a while, when can we expect a complete fix?

Daniel Veillard 2018-05-23 13:40:13 UTC Comment 36

The fix is commited in git, it was pushed in last release, there is no information that anything more is needed, with current reelease, if not
please post the problem here,

thanks,

Daniel

Nikunj Mavani 2018-05-24 09:11:22 UTC Comment 37

Hi Denial,

Are we getting complete fix of vulnerabilities of the CVE in libxml2-2.9.8?

If yes, why this bug status is "Reopened"?

Thanks,
Nikunj Mavani

Nikunj Mavani 2018-05-31 07:18:40 UTC Comment 38

Hi Denial,

Could you please revert to the queries mentioned in comment#37?

Thanks,
Nikunj

Nikunj Mavani 2018-06-14 08:30:27 UTC Comment 39

Could you please revert to the queries mentioned in comment#37?

Thanks,
Nikunj

Tobias Mueller 2018-08-17 15:34:27 UTC Comment 40

Okay, this seems to be resolved indeed. Please reopen if this continues to be a problem.

tanaya patil 2018-11-26 10:28:59 UTC Comment 41

Can anyone please specify the version in which this issue is fixed or mention it in the "target milestone" field?

David Kilzer 2018-11-26 18:36:31 UTC Comment 42

```
$ git tag --contains ad88b54f1a28a8565964a370b5d387927b633c0d
v2.9.8
v2.9.8-rc1
```

tanaya patil 2018-12-05 06:58:24 UTC Comment 43

Would like to know if building libxml2 v2.9.8 is dependent on some gcc version.
error while building:

```
  CC       SAX.lo
cc1: error: unrecognized command line option "-Wno-array-bounds"
make[2]: *** [SAX.lo] Error 1
```

gcc version used: gcc 4.2

if it is dependent on gcc version, can there be a workaround to make this work? as upgrading gcc is not a feasible choice for me.

Ravi Sarda 2019-04-23 11:00:37 UTC Comment 44

I am facing the same issue as mentioned in Comment 43 any suggestion or workaround?

Ravi Sarda 2019-04-23 11:02:13 UTC Comment 45

will removing the flag help?

https://bugs.eclipse.org/bugs/show_bug.cgi?id=422741