

[New issue](#)

MTU length not validated: possible call to alloc() or realloc()... with size 0 #390

[Closed](#)

j4kb4dw0lf opened on Apr 28, 2025 · edited by j4kb4dw0lf

Edits ▾ ⋮

- Hardware description: x86_64 PC, standard desktop
- OS: Ubuntu 22.04 LTS
- Installation type: Built from source, CMake default flags, FastDDS 2.14
- Version or commit hash: 3.0.1 (main branch)

Steps to reproduce the issue

Force the MTU length parameter to be zero during client creation.

Trigger an event where an allocation for a message buffer is attempted or a realloc on one is done, e.g. a response from the agent to a status message.

Expected behavior



The agent should detect invalid MTU values (e.g., ≤ 0) **before** calling memory allocation functions, and either reject the setup with a clear error or default to a minimal safe value.



Actual behavior

When the MTU length is not correctly checked, the alloc function or the realloc function can be called with a parameter of 0. This results in undefined behavior depending on the system's allocator and could lead to crashes, hard-to-debug issues, or memory corruption risks.

Additional information

Proper validation on the MTU field would prevent this behavior. Suggested fix: check that MTU is greater than 0 before calling any allocation functions and assert or log error otherwise.

  **j4kb4dw0lf** changed the title ~~MTU length not validated: possible call to alloc() with size 0~~ MTU length not validated: possible call to alloc() or resize() with size 0 on Apr 28, 2025

  **j4kb4dw0lf** changed the title ~~MTU length not validated: possible call to alloc() or resize() with size 0~~ MTU length not validated: possible call to alloc() or realloc()... with size 0 on Apr 28, 2025

 **4ntn** on May 6, 2025 ...

Hello [@j4kb4dw0lf](#),

Thank you for pointing this out.

Can you specify in what lines the alloc is occurring, and how you are setting the MTU?

I don't understand exactly what you are referring to.

 **j4kb4dw0lf** on May 6, 2025 · edited by j4kb4dw0lf Edits ▾ Author ...

Hello [@4ntn](#), there is no way for me to refer to a specific line, if i were to tell you one in particular it would probably need to be here:

[Micro-XRCE-DDS-Agent/include/luxr/agent/message/OutputMessage.hpp](#)

Line 34 in [155cfaa](#)

```
34      : buf_(new uint8_t[len]{0}),
```

where the buffer for output messages are allocated.

This is done in various instances of processing of submessages received by the agent (specifically in the `push_output_submessage` subsequent call).

The "setting" of the MTU length is actually part of a university project about software security where we were tasked to find potential security issues and bugs in the Agent, the campaign resulted in the two bugs i reported as issues.

The MTU length is specifically the one that is deserialized from a `client_representation` sent in a `CREATE_CLIENT` submessage (the last 2 bytes of said submessage).

In order to patch this issue and prevent crashes i would advise the approach of either setting a minimum mtu length value or adding additional checks to the `session_info_.mtu` used here:


[Micro-XRCE-DDS-Agent/include/luxr/agent/client/session/Session.hpp](#)

Line 272 in [155cfaa](#)

```
272      inline bool Session::push_output_submessage(
```

  **4ntn** mentioned this [on May 8, 2025](#)

 [Validate that received MTU is greater than 0 on client creation #392](#)


 **4ntn** on May 8, 2025 ...

Hello again, [@j4kb4dw0lf](#) ,

I have been able to reproduce the issue.

I have implemented this fix: [#392](#)



It would be nice if you could test it in your setup to confirm.

 **j4kb4dw0lf** on May 8, 2025 Author ...

Hello [@4ntn](#) and thank you for the prompt answer and fix, i can indeed confirm the patch you provided fixes the issue addressed. i am thrilled to have been helpful in the development of the Agent, i will also "exploit" this little moment of glory to let you know the other issue i opened in this repo [#389](#) could require an even easier fix and is equally as potentially harmful as this one you just worked on fixing! thank you again

 **4ntn** on May 12, 2025 ...

I'm closing this now, thank you for your contribution, [@j4kb4dw0lf](#) !

  **4ntn** closed this as [completed](#) [on May 12, 2025](#)

[Sign up for free](#) to join this conversation on **GitHub**. Already have an account? [Sign in to comment](#)

Metadata

Assignees

No one assigned

Labels

No labels

Type

No type

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

No branches or pull requests

Participants

