

Getting your Code into Wireshark Releases and Latest Additions to the Wireshark API

June 16th, 2009

Michael Tüxen

Professor / Wireshark Core Developer | Münster University of Applied Sciences

SHARKFEST '09

Stanford University

June 15-18, 2009

Outline

- Overview on additions to the API.
- Disclaimer.
- General hints for getting your code into the official distribution.
- Specific hints...
- Case studies.
- Conclusions.

GLib and GTK+

- GLib 1.* and GTK+ 1.* support is gone!
- GLib 2.4.0 or higher required.
- GTK+ 2.4.0 or higher required for Wireshark.
- Latest stable releases:
 - GLib 2.20.3
 - GTK+ 2.16.2
- Native Mac OS X (Aqua)?

Private Things...

- `proto_mark_private();`
- `proto_is_private();`
- Checked when doing remote queries.

Hidden Things

- `proto_tree_add_*_hidden()` is deprecated. You can use `PROTO_ITEM_SET_HIDDEN()` if needed.

Simplifications

- We no longer need to check for (tree != NULL) when using proto_tree_add_*()
- We no longer need to check for a column when using col_add_*() and col_set_*()

Dynamic Strings with Packet Lifetime

- `ep_strbuf_new`, `ep_strbuf_new_label()`,
`ep_strbuf_sized_new()`
- `ep_strbuf_append_vprintf()`,
`ep_strbuf_printf()`, `ep_strbuf_append_printf()`,
`ep_strbuf_append()`, `ep_strbuf_append_c()`,
`ep_strbuf_truncate()`

Dynamic Strings with Capture Lifetime

- `tvb_get_seasonal_string()`,
`tvb_get_seasonal_stringz()`,

A new platform...

- 64-bit Windows...
- The buildslave runs Windows XP 64-bit.
- Windows is LLP64.
- Others systems often are LP64.
- `size_t` is 64-bit, `long` and `unsigned long` is 32-bit.
- Casts are needed...

Disclaimer

- I'm not related to CACE technologies.
- I'm not Gerald.
- I'm just one core developer.
- The following is mostly my opinion...
- If you disagree, please speak up!

Why to contribute?

- Writing code / debugging code is very time consuming.
- Benefits you get from contributing include:
 - Get others to test your code.
 - Get others to improve your code.
 - No effort for code maintenance.
 - No effort for code distribution / application distribution.

Core Developer

- Someone with the commit bit.
- It is only one repository.
- No specific area of responsibility.
- Status seems to be permanent.
- About 41 people listed at <http://wiki.wireshark.org/Developers>

How to contribute

- Provide a bug report at the bug tracker <https://bugs.wireshark.org/bugzilla/>
- Provide a patch using the bugtracker available at <https://bugs.wireshark.org/bugzilla/>
- Discuss things at the developers mailing list wireshark-dev@wireshark.org

Some General Hints...

- Read doc/README.developer.
- Base your code on the development branch.
See <http://www.wireshark.org/develop.html>
- Don't change lines you do not want to change.
- Adopt to coding style in the files you are changing.
- Test you change. And provide the possibility for core developers to test...

Some Specific Hints

- Do not use C++ code.
- Do not assume that your platform is the only platform.
- Use generic `proto_tree_add_item()` when possible.
- Be careful when allocating memory and accessing it.
- Look at the already existing code.

Case Studies

- A general bug report on a dissector.
- A FreeBSD specific bug.
- A GUI related new feature requiring remote access.
- An SCTP bug.
- A recent pcapng bug.

Lessons Learned

- Use the bug tracker and the developer mailing list.
- Resolve technical problems in a timely way.
- Try to make the job for the core developers as easy as possible.
- Get the attention of a core developer.
- Be patient, be insistent.
- Communication is very important.