Wireshark User Interface

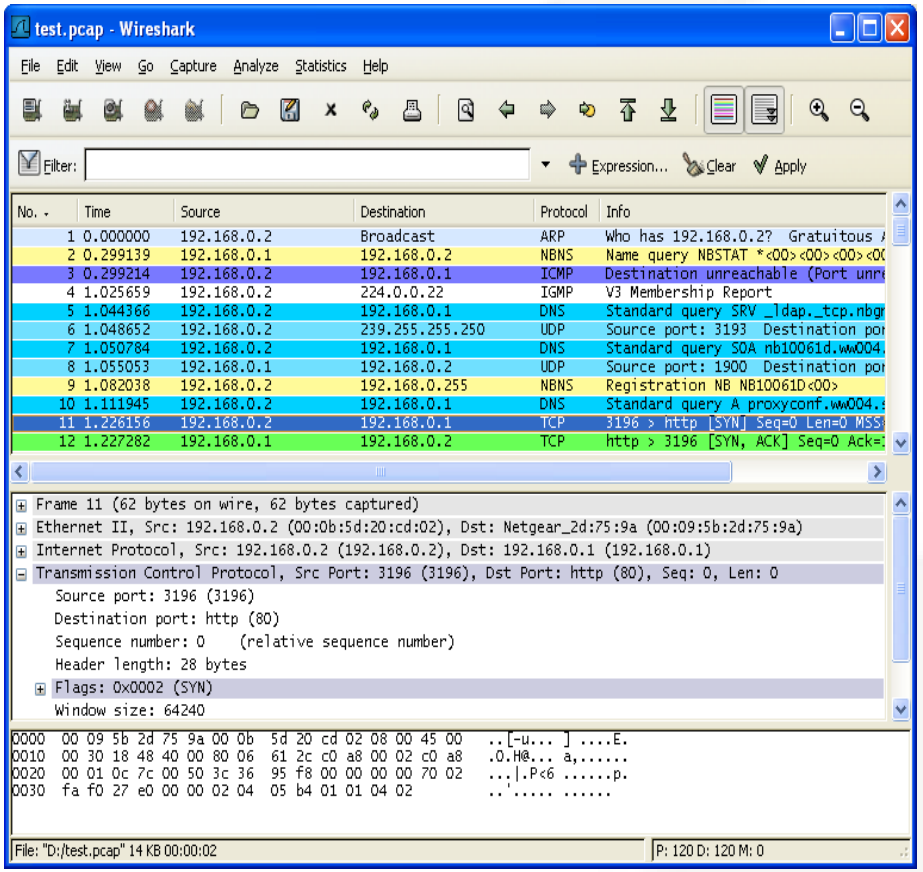
Programming and Extending
April 1, 2008

Ulf Lamping
Wireshark Core Developer

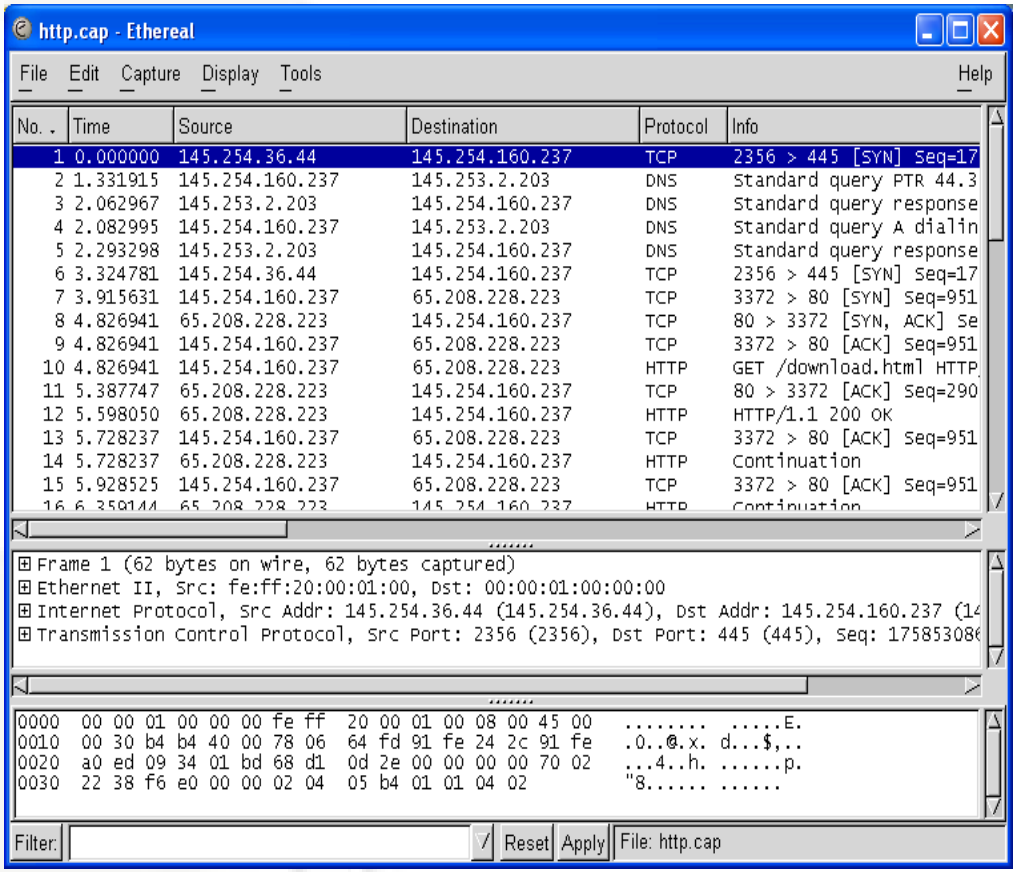
SHARKFEST **'08**
Foothill College
March 31 - April 2, 2008

Now and then ...

2008: Wireshark 1.0.0



2003: Ethereal 0.9.10a



Agenda

Introduction

GTK+

GUI code

Protocol Specific

Various

Introduction

So let's start ...

Reasons to work on the WS GUI

Add GUI stuff for “your” protocol

Add a cool new feature

Enhance existing stuff

Fix that GUI behavior that permanently annoys you

... or whatever comes to your mind – it's open source!

Prepare WS GUI Development

As if you would develop a Wireshark dissector!

Install a Wireshark development environment

- Wireshark Developers Guide

Get in touch with general Wireshark development

- doc/README.developer

Tip: A basic understanding of Wireshark development will help a lot to start any GUI development!

GTK+

The GIMP ToolKit

What is GTK+?

A Toolkit to build GUI's

Platform independent (Unix, Windows, MAC OS X, ...)

Widely used (GNOME, GIMP, Inkscape, ...)

Mature (10+ years)

Written in ANSI-C

A set of libraries

GTK+ Libraries

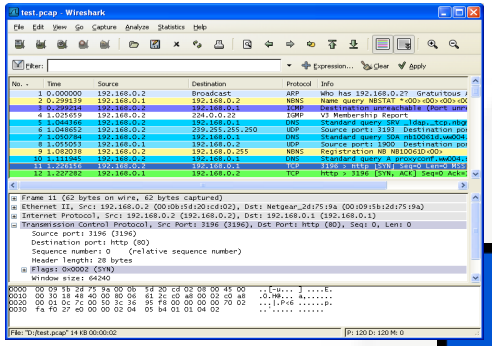
GTK+

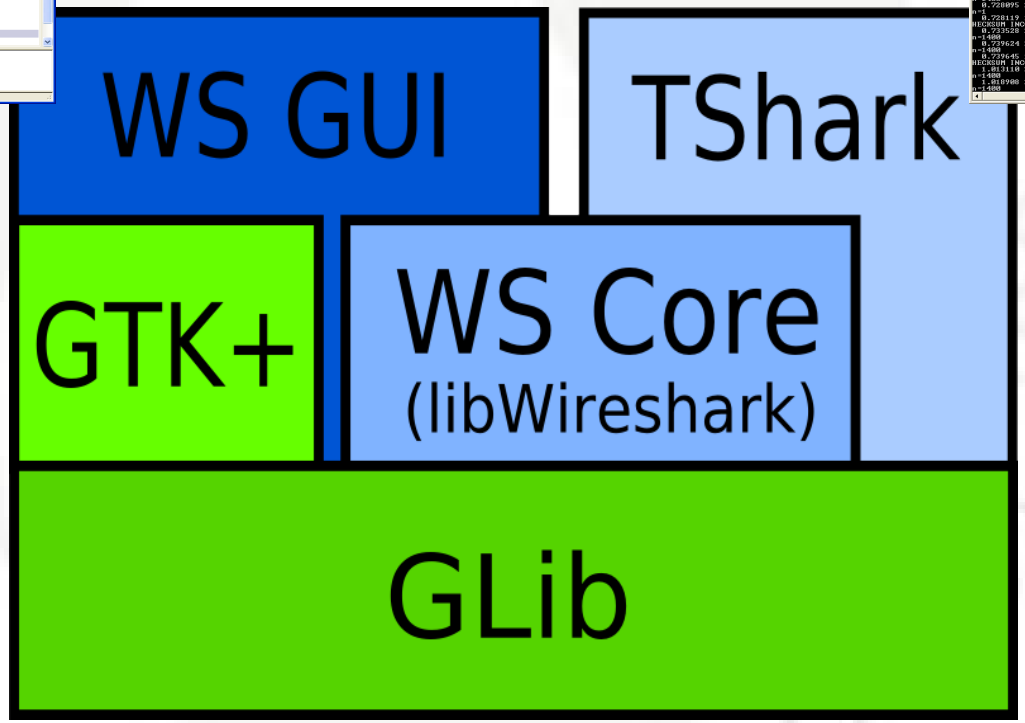
- **GTK+ – Widgets** — ***gtk_container_add()***
- GDK – Low level graphics — *gdk_draw_line()*
- Pango – Font rendering — *pango_layout_set_text()*
- GdkPixbuf – Images — *gdk_pixbuf_get_width()*
- ATK – Accessibility — *atk_text_get_text()*

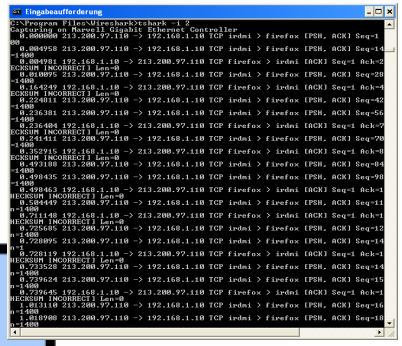
GLib

- **Glib – Portability basics** — ***g_list_new()***
- Gobject – Object base — *g_object_set()*

Wireshark vs. Tshark library usage







Wireshark code

Common for all

- uses Glib
- ~1,500,000 lines of code together (most is protocol specific)
- plain ANSI-C (“99.9%” platform independent)

Tshark don't use GTK+

- uses Glib
- ~ 11000 lines of code (lot's of code is protocol specific)

Wireshark GUI code

- uses Glib, GTK+, ...
- ~ 75000 lines of code (lot's of code is protocol specific)

GTK+ “ancient” major Version V1

No active development / no bug fixing

- Last GTK+ V1.3 release in 2004

Current GTK+ versions still support 1.x functions

- ... at least till today!
- BUT: A lot of functions are marked as deprecated in V2.x

Wireshark still works with V1.2 / V1.3

- “Fallback” for Wireshark users if GTK V2.x fails
- Some new WS features won't appear if V1.x based
- Development / Maintenance burden!

GTK+ “current” major Version V2

A lot more features – compared to V1.x – and still rising

- Example: Enhanced Text display (color, underlined, ...)

The minor version “even numbers” of V2.x are stable

- x=0, 2, 4, 6, 8, 10, 12, 14

Under active development

- However, bug fixing may take a while

“Common platform” for Wireshark

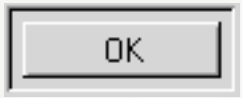
GTK+ Version Differences

Features new in GTK V2.x will not even compile on V1.x

```
#if GTK_MAJOR_VERSION >= 2
/* some cool new stuff */
#else
/* "fallback" for GTK+ V1.x - often just empty */
#endif
```

Lot's of common definitions in *gtk/compat_macros.h:*

BUTTON_NEW_FROM_STOCK*(GTK_STOCK_OK);*

- V1.x: without icon 

V2.x: with icon 

GTK+ “future” major Version V3

Probably breaks compatibility with older versions

First V3 release is still far away

- currently GTK community discussions

V3 is too far away for current Wireshark development!

- No need to hurry here!

Glade – GTK+ prototyping

Pros

- “Play” with dialog layouts
- Graphically “click together”
- Learn GTK+ widget layout
- Much faster than C coding

Cons

- Currently not suitable for WS productive code!
- Interfacing to WS code is not possible!
- Hard to find Win32 binary

GTK+ Summary

GTK+: A Platform independent (GUI) framework

A lot of WS GUI code depends on GTK+

WS support of V1.x might be dropped sooner than later

GTK+ V3.x still far away

Tip: Read the GTK+ Tutorial to get a good GTK+ start!

Wireshark GUI Code

"Sniffing the glue,
that hold's the Wireshark together"

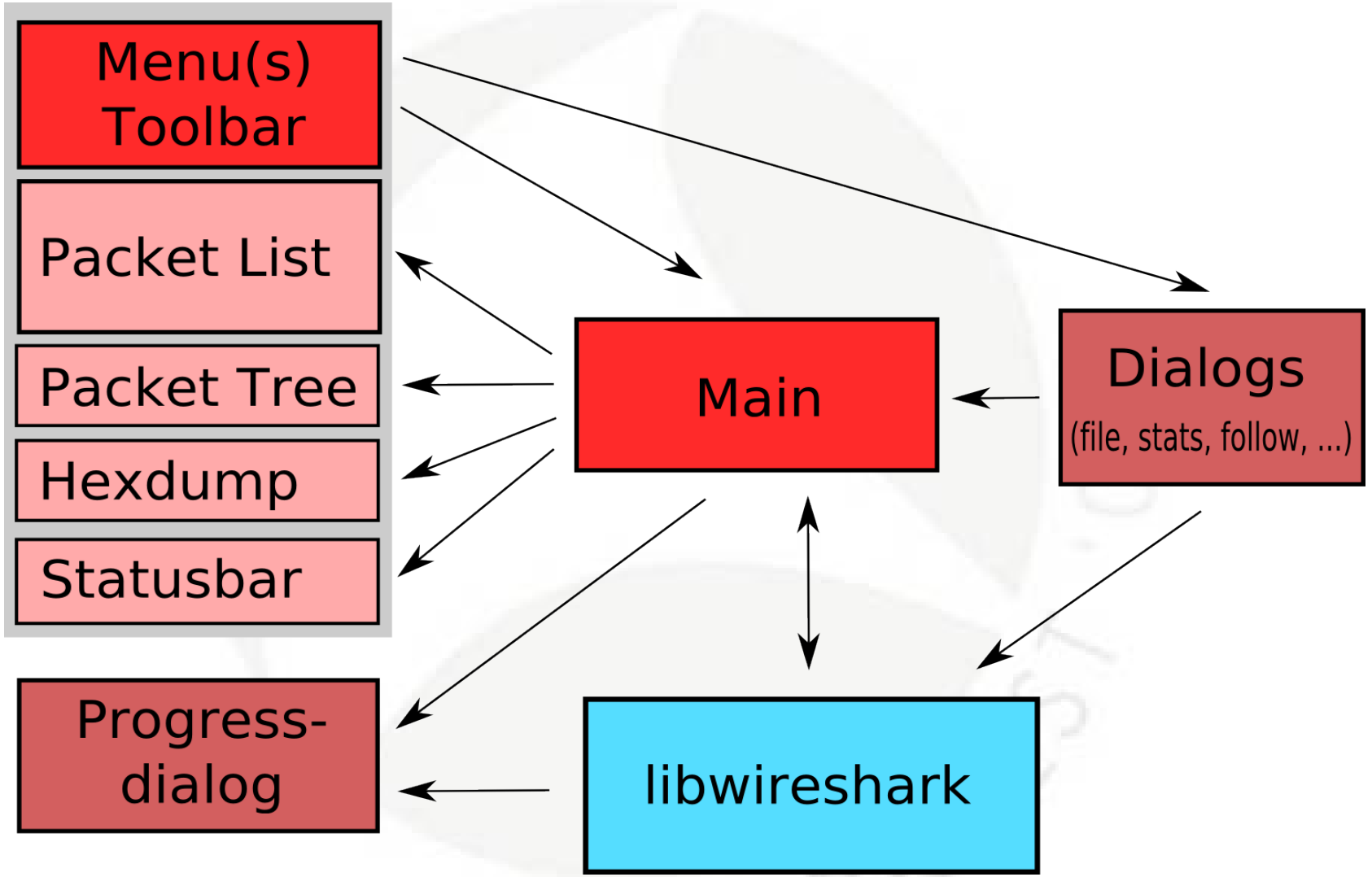
Wireshark GUI code documentation

The code documentation is pretty sparse :-(

- GTK+ API itself is well documented
- Lot's of code examples in Wireshark
- Find a simple / suitable example – and go!

The following hopefully helps to get a good start ...

Major GUI Blocks (simplified)



Where to find the GUI sources?

GTK+ specific stuff is in the *gtk* dir ;-)

- Contains both V1 and V2 stuff (most code is mixed)

Do not edit anything in the *gtk2.tmp* dir!

- This stuff is generated (copied) by the build process

“Secret” filename conventions in gtk/*

**_prefs (single pages)*	Edit/Preferences
*follow_**	Analyze/Follow ... Stream
*expert_**	Analyze/Expert Info
*conversations_**	Statistics/Conversations
*hostlist_**	Statistics/Endpoints
**_stat*	Statistics/... (various)
**_dlg*	Various dialogs
**_utils*	GUI utility code without visible components

“Prominent files” (in the gtk dir)

main.c	Main loop & lot's more (command line, ...)
menu.c	Application and context menus
packet_list.c	Packet List
proto_draw.c	Packet Details and Hexdump

Helpers:

gui_utils.h	Windows creation and alike
dlg_utils.h	Dialog related (but also look at *gui_utils.h*)
file_dlg.h	Generic file open / save

Menu

gtk/menu.c

same mechanism for application and context menus

factory based:

- ITEM_FACTORY_STOCK_ENTRY(...)
- ITEM_FACTORY_ENTRY(...)

add stock items

- gtk/compat_macros.h & toolbar.c (& .xpm icons)

“grey out" menu stuff currently not available

- set_menu_sensivity(),e..g. set_menus_for_capture_files()

Specialties

Platform independent, but:

- Windows Common dialogs (open, print, ...)

Window vs. Dialog

- Window: showing data
- Dialog: Asking question(s), Progress, ...

Dialog Buttons

- Win32: OK / Apply / Cancel vs. GNOME: Cancel / Apply / Ok

Protocol specific GUI

Taps, Statistics, ...

Taps

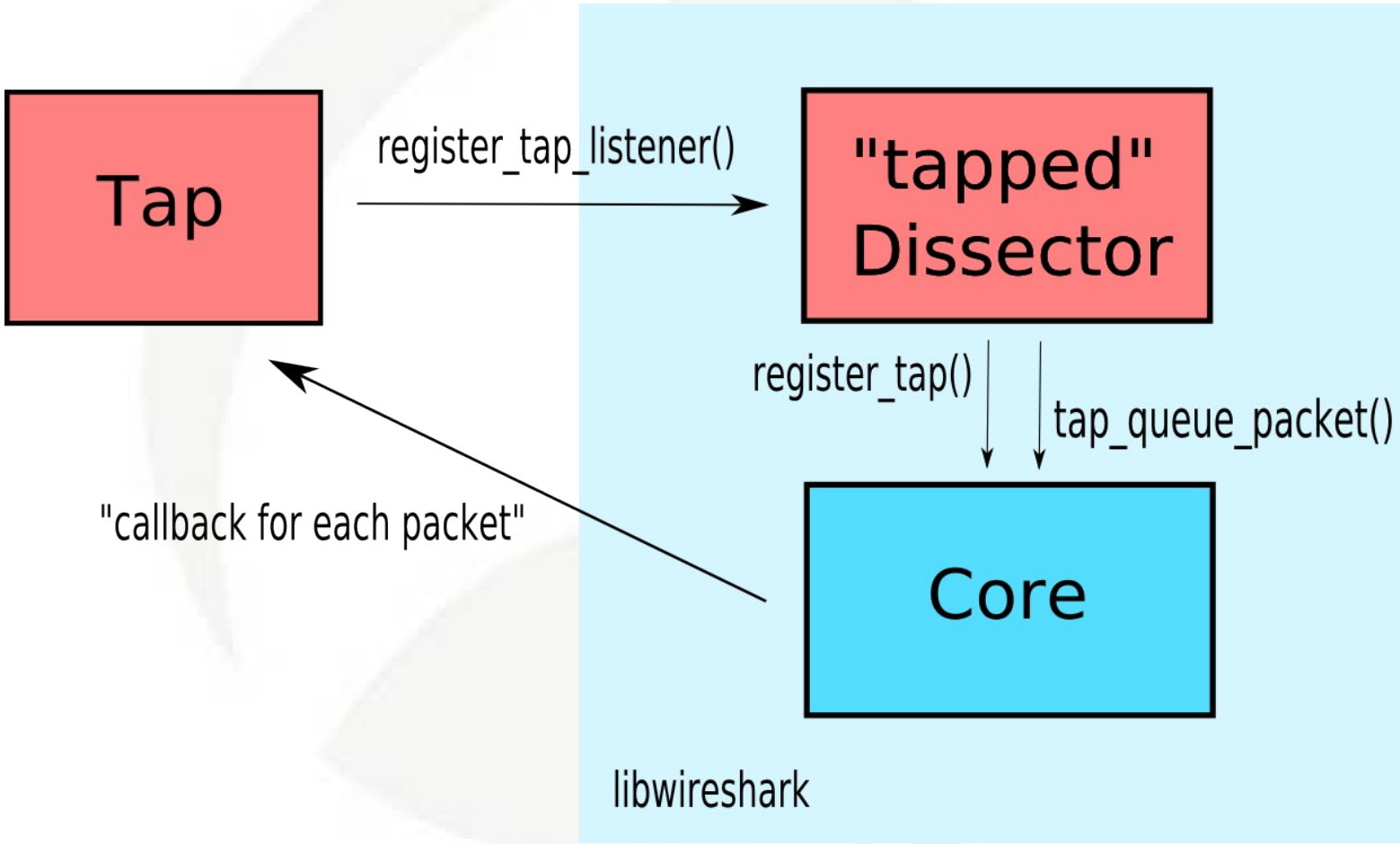
An interface to libwireshark

Get a callback for each protocols packet

Simple to implement

... see *doc/README.tapping* for details

Tapping mechanism



Prepare the “tapped” dissector

#include <epan/tap.h>

static int <protocol>_tap = -1;

... in proto_register_<protocol>():

<protocol>_tap = register_tap("<tapname>");

... “somewhere” in dissect_<protocol>:

if(have_tap_listener(<protocol>_tap)

tap_queue_packet(<protocol>_tap, pinfo, <optional pointer>);

Tap Listener

void register_tap_listener_<protocol>(void) {

register_tap_listener(

<tapname> - the taps name from register_tap()

<tapdata> - “your data pointer” for packet_cb()

filterstring – optional “display filter” for packet callback (can be slow!)

reset_cb – callback: new file (e.g. reset the stats)

packet_cb – callback: new packet coming in

draw_cb – callback: (re)draw the display

)

}

... and add your file to *WIRESHARK_TAP_SRC* in *gtk/Makefile.common*!

Statistics

Build your protocol statistics from scratch? No!

- Various examples are in the code!
- Generic helper code is available!

Avoid code duplication

- maintenance nightmare!
- “carve out” common code and provide a generic interface

Statistics helper (simple)

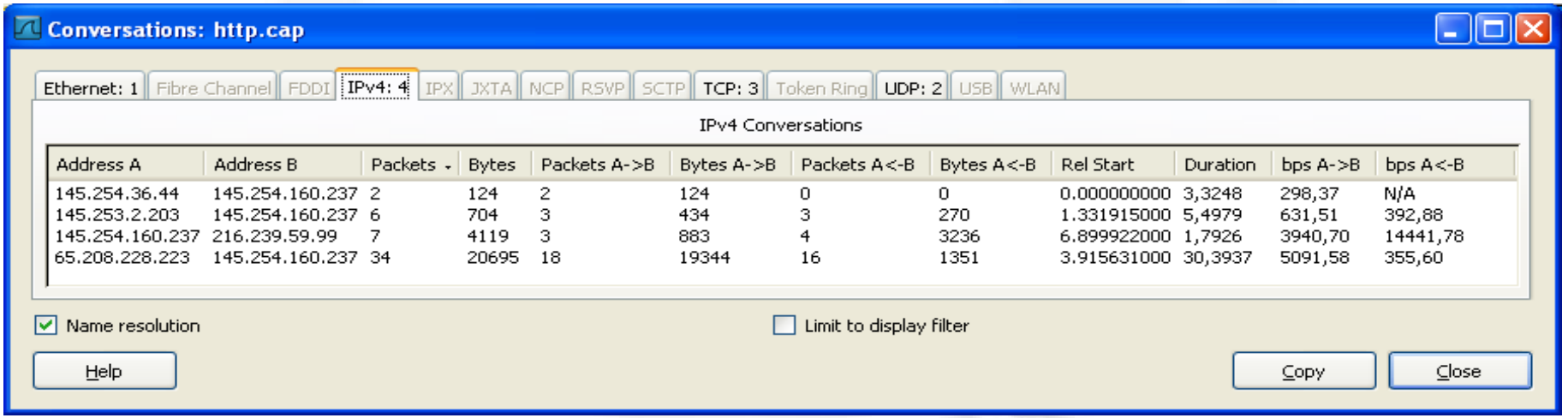
For these: no GTK+ knowledge required!

conversations_table.h

- conversations_fc.c (91LOC)

hostlist_table.h

- hostlist_fc.c (96 LOC)



Statistics helper (advanced)

service_response_time_table.h

- fc_stat.c – 219 lines of code

graph_analysis.h

- flow_graph.c – 692 lines of code

follow_stream.h

- follow_udp.c – 300 lines of code

Various

Might be interesting ...

Usability

Classical usability problems:

- Exit a program with: “Save the changes?” Yes / No / Cancel
  - What happens if you click Cancel?
- This dialog is ugly! How can I improve it?
- When to use radio buttons and when a drop down box?

Find help in the:

“GNOME Human Interface Guidelines”

Pitfalls

Long running tasks

- Call main_update() frequently to update the GUI
  - ... at least once a second
- add a progress bar (see *gtk/progress_dlg.h*)!
  - ... keep the user informed

Don't use multithreading in the GUI!

- ... or to be more precise: in general WS development!

On the Web ...

GTK+ Tutorial:

http://library.gnome.org/devel/gtk-tutorial/stable/

GTK+ documentation:

http://www.gtk.org/documentation.html

Glade: http://glade.gnome.org/

GNOME Human Interface Guidelines:

http://developer.gnome.org/projects/gup/hig/2.0/index.html

Wireshark developer's mailing list:

http://www.wireshark.org/mailman/listinfo/wireshark-users

Summary

The GUI is no rocket science!

Do some dissector development first!

Learn GTK+ with the GTK+ Tutorial

Usability guidelines help!

... and then:

Use the source Luke!

Hopefully we'll see some nice new features :-)

Questions?

??

??

??

??

??

??

??

??

??

??

Thanks for your attention!

ulf.lamping@web.de