# Wireshark Developer and User Conference

## Writing Wireshark Dissectors & Plug-ins

June 14, 2011


**Gerald Combs**
Wireshark Project Creator

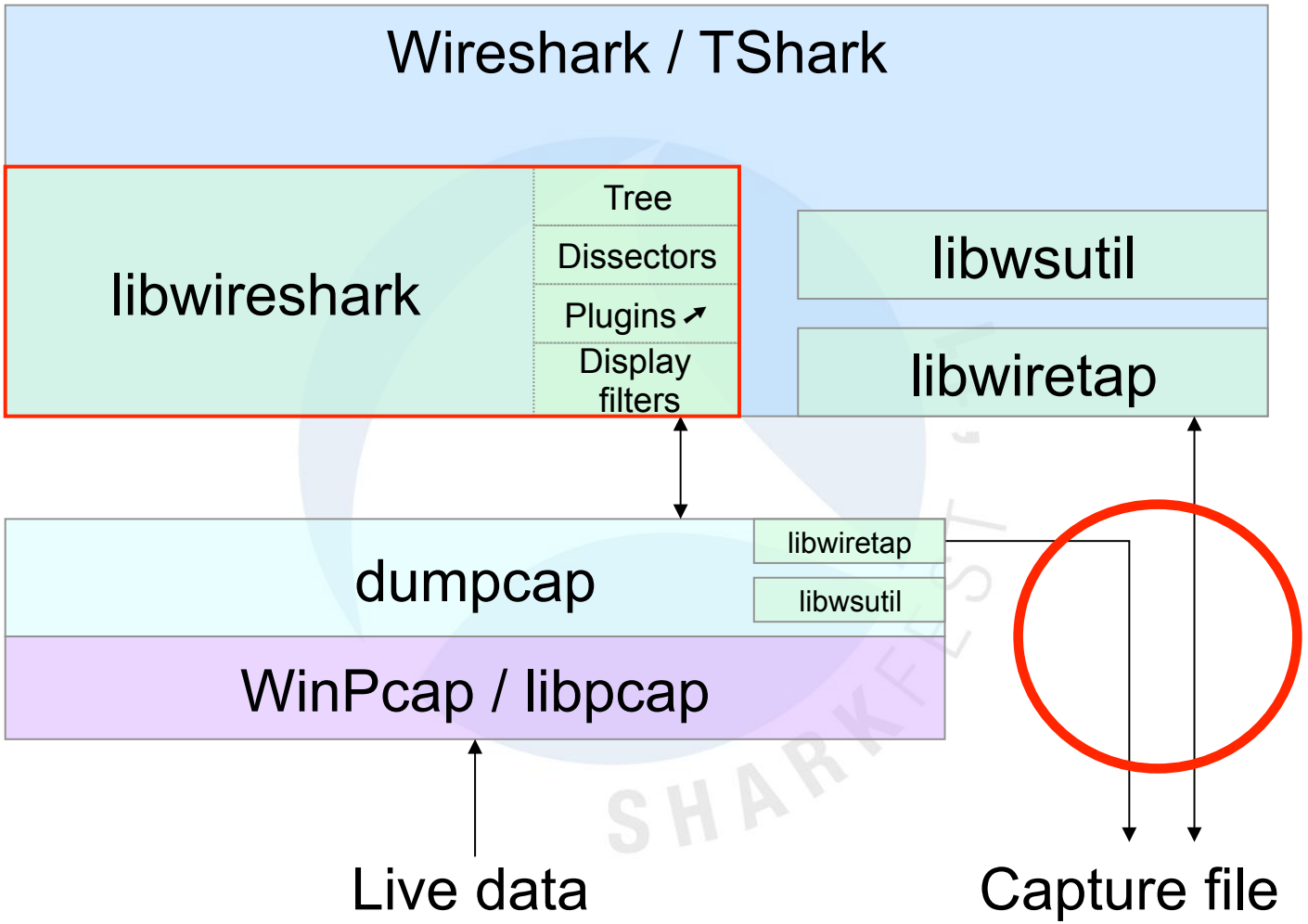
**Stephen Fisher**
Wireshark Core Developer


**SHARKFEST '11**
Stanford University
June 13-16, 2011

# Wireshark Development

- Distributed
- Plain old boring C
- Multi-platform
- Multi-interface
- GPL 2-or-later
- SVN
- Brisk development pace

# Application Architecture



# Build Requirements

- Source code (of course)
- Visual C++ || gcc || SunPro C++ (|| LLVM?)
- Libraries: WinPcap/libpcap, GLib, GTK+, zlib, GNUTLS, c-ares, libsmi, …
- Support tools: Python, Perl, Linux/UNIX shell

# Build Requirements - Windows

- Visual C++ 6.0 to 2010
- Python 2.4+ (No 3.0 yet)
- Optional: NSIS, TortoiseSVN
- Not optional: Cygwin

# Getting the Code

- Subversion

  http://anonsvn.wireshark.org/wireshark/trunk

  http://anonsvn.wireshark.org/wireshark/trunk-1.6

- Tar files

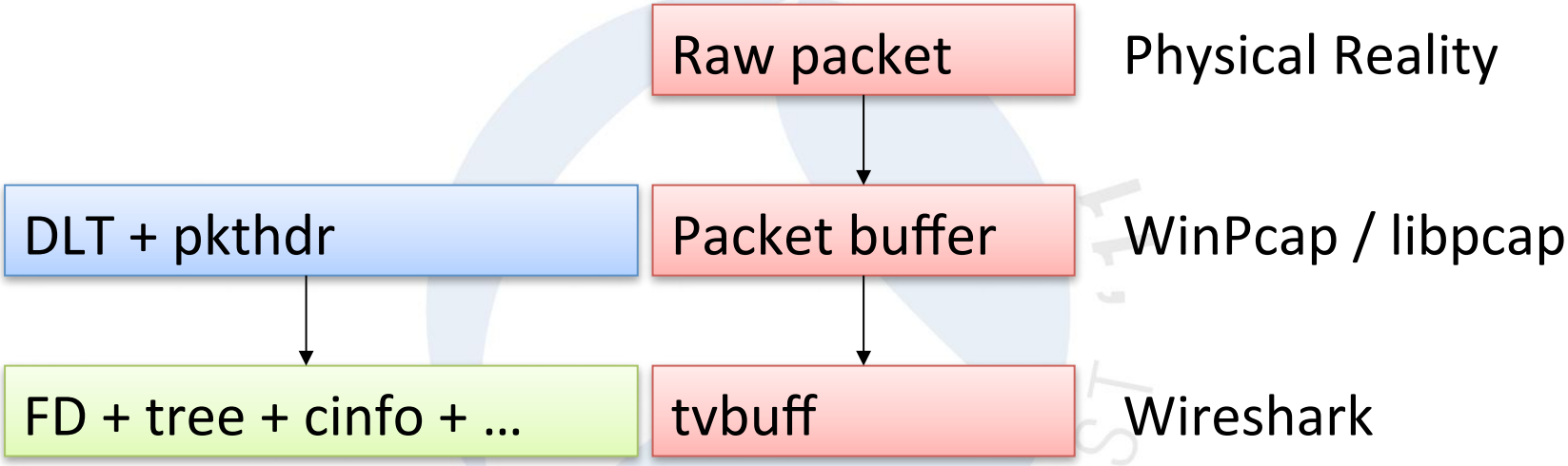
  http://www.wireshark.org/download/src

- Want to send us a patch? `svn diff`

# Source Directory Overview

| | |
|---|---|
| *root* | CLI applications, common code |
| doc | READMEs, man pages |
| docbook | Guides |
| epan | Dissection |
| dissectors | Built-in dissectors |
| gtk | User interface |
| packaging | Platform installers |
| plugins | Plugin dissectors |
| wiretap | File formats |
| wsutil | Shared utility routines |

# Packet Data + Metainformation



# Core Data Structures

- You get:
  - tvbuff: Protocol data access
  - packet_info, frame_data: Packet meta-info
  - proto_tree: Detail tree. Sometimes.
- You provide:
  - header_field_info

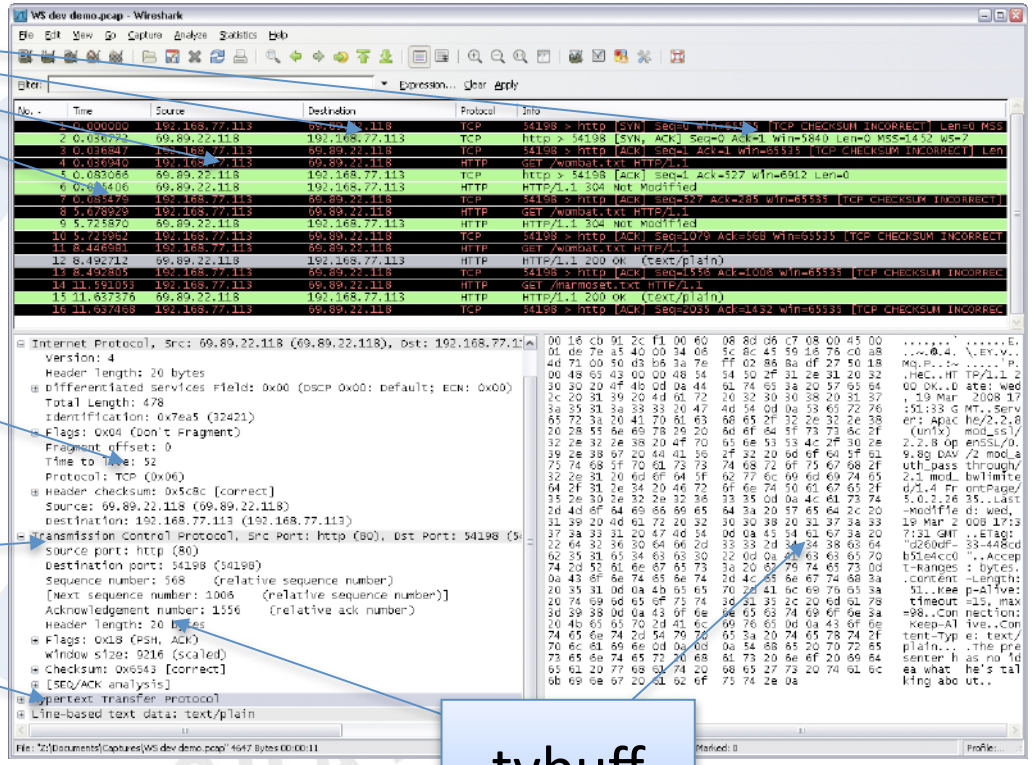
# UI Element Origins

col_set_str()

proto_tree_add_*()

hfinfo

dissector_add_uint()



# Getting Called

- Normal

  ```
  dissector_add_uint /* Changed in 1.6 */
  ```

- Heuristic

  ```
  heur_dissector_add
  ```

- On the fly

  ```
  dissector_add_handle
  find_dissector + call_dissector
  ```

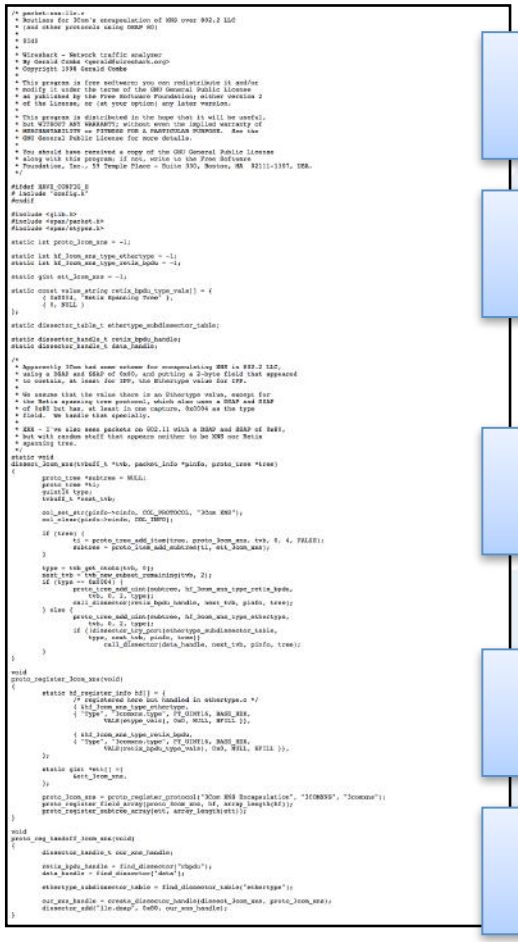
# DNS Dissection Call Stack

```
dissect_dns_udp()          /* packet-dns.c */
decode_udp_ports()         /* packet-udp.c */
dissect_udp()              /* packet-udp.c */
dissect_ip()               /* packet-ip.c */
dissect_eth_common()       /* packet-eth.c */
dissect_frame()            /* packet-frame.c */
dissect_packet()           /* Add top-level structs */
process_packet()           /* DLT from wiretap */
main()
```

# When Do You Get Called?

- File load
- User clicks on packet
- Name resolution
- Lots of other times

# Parts of a Dissector



Legal

Globals (value strings)

Dissection Routines

Registration Routines

Header Field Info

# Registration

packet-arp.c

```
void
proto_register_arp(void)
{

  proto_arp = proto_register_protocol("Address Resol
                                      "ARP/RARP", "a

}

void
proto_reg_handoff_arp(void)
{
  dissector_handle_t arp_handle;

  arp_handle = find_dissector("arp");

  dissector_add("ethertype", ETHERTYPE_ARP, arp_hand
  dissector_add("ethertype", ETHERTYPE_REVARP, arp_h
  dissector_add("arcnet.protocol_id", ARCNET_PROTO_A
  dissector_add("arcnet.protocol_id", ARCNET_PROTO_A
  dissector_add("arcnet.protocol_id", ARCNET_PROTO_R
}
```

register.c

```
void
register_all_protocols(register_cb cb, gpointer clie
{
  {extern void proto_register_1722 (void); if(cb) (*

  {extern void proto_register_arp (void); if(cb) (*c

  {extern void proto_register_zrtp (void); if(cb) (*
}

void
register_all_protocol_handoffs(register_cb cb, gpoin
{
  {extern void proto_reg_handoff_1722 (void); if(cb)

  {extern void proto_reg_handoff_arp (void); if(cb)

  {extern void proto_reg_handoff_zrtp (void); if(cb)
}
```

# tv_buff: Testy, virtual buffers

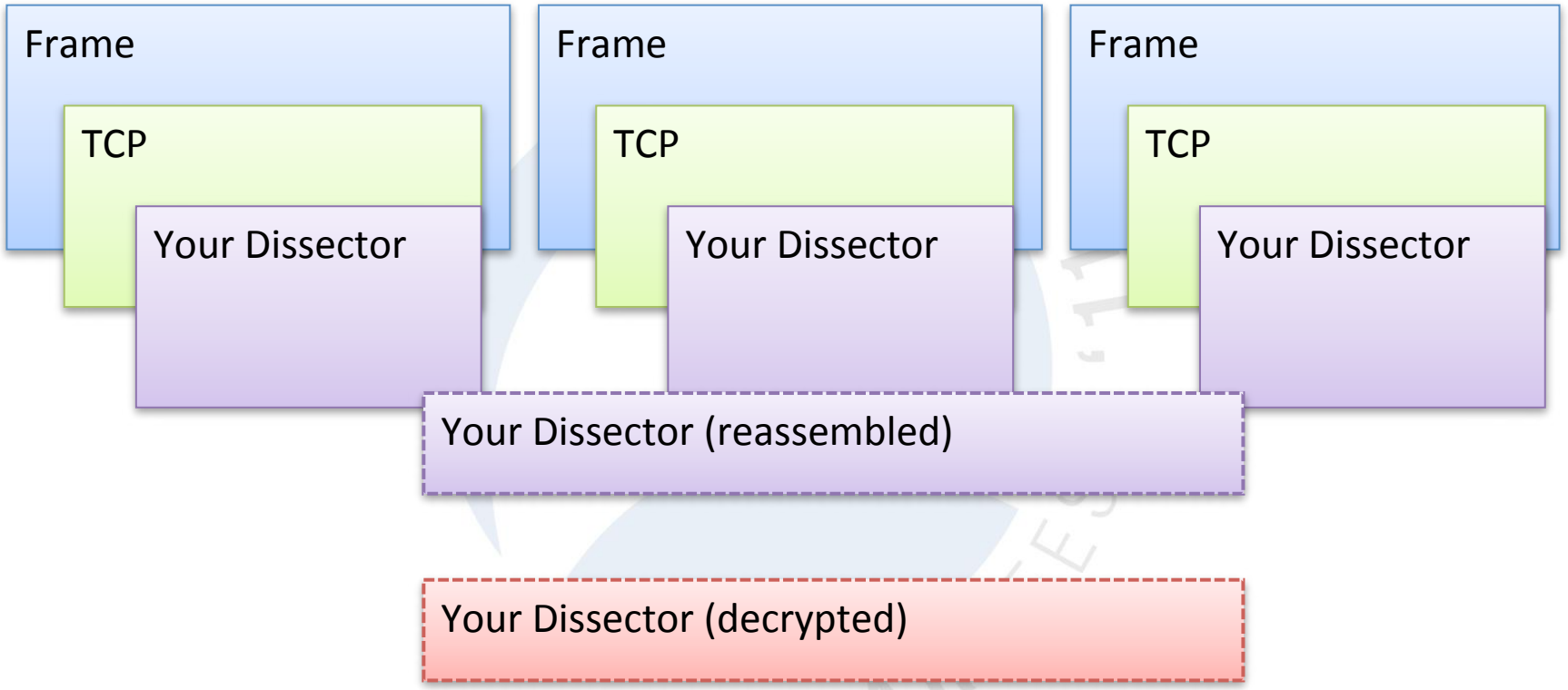
- Data buffers & extraction
- `tvb_get_…`
  - `guint8`
  - `ntohs, ntoh24, ntohl, ntoh64`
  - `letohs, …`
  - `ipv4, ipv6`
  - `string, …`
- `tvb_mem…`

Make your own!
Impress your friends!

epan/
tvbuff.h

Chain them
together!

# tv_buff Examples



# packet_info & frame_data

- High and low-level metainfo
- epan/packet_info.h, epan/frame_data.h
- Frame data: Wire information
  - Length, timestamps, DLT
- Packet Info: State information
  - Addresses, ports, reassembly, protocol data

# header_field_info

- Describes protocol elements
- Data type, filter name, descriptions
- Enums - Value/Range/TF Strings
- epan/proto.h
- Unique

# hfinfo Examples

```
{&hf_ieee80211_ff_block_ack_timeout,
  {"Block Ack Timeout", "wlan_mgt.fixed.batimeout",
  FT_UINT16, BASE_HEX, NULL, 0, NULL, HFILL }},


{&hf_smb_file_type,
  { "File Type", "smb.file_type", FT_UINT16, BASE_DEC,
  VALS(filetype_vals), 0, "Type of file", HFILL }},


{&hf_ieee80211_ff_block_ack_params_amsdu_permitted,
  {"A-MSDUs", "wlan_mgt.fixed.baparams.amsdu",
  FT_BOOLEAN, 16, TFS (&ff_block_ack_params_amsdu_permitted_flag),
  0x0001, "A-MSDU Permitted in QoS Data MPDUs", HFILL }},
```

# proto_tree

- Detail tree (middle pane)
- Might be NULL
- epan/proto.h
- Can be hidden or “generated”
- Avoid proto_tree_add_text()

# Adding Tree Items

- Names
- Data type
- Tree position
- Packet position
- New in 1.4: Last field = encoding

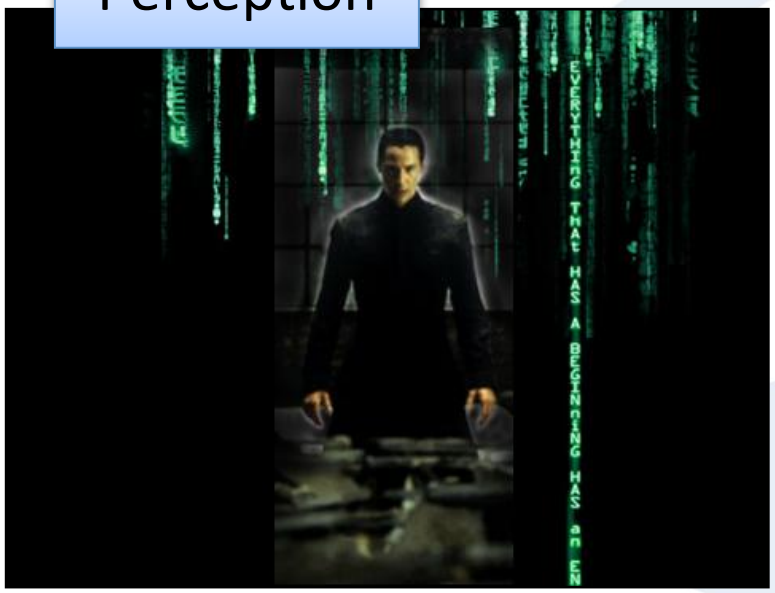
```
proto_item   *ti;
proto_tree   *sub_tree = NULL;

ti = proto_tree_add_item(tree, …);
sub_tree = proto_item_add_subtree(ti, …);

proto_tree_add_item(sub_tree, …);
proto_tree_add_uint_format(sub_tree, …);
```

# Naïve Dissection

```
typedef struct _my_pdu_header_t {
    guint8 version;
    guint16 type;
    guint16 code;
    guint16 len;
} my_pdu_header_t;


    my_pdu_header_t *hdr;

    hdr = (my_pdu_header_t *) tvb_get_ptr(tvb, 0, sizeof(my_pdu_header_t));

    proto_tree_add_text(tree, tvb, 0, 1, "Version: %u", hdr->ver);
    proto_tree_add_text(tree, tvb, 1, 2, "Type: %u", hdr->type);
    proto_tree_add_text(tree, tvb, 3, 2, "Code: %u", hdr->code);
    proto_tree_add_text(tree, tvb, 5, 2, "Len: %u", hdr->len);
```

# tvb_get_ptr()

Perception



Reality

Image sources:
http://www.sitevip.net/the_matrix/images/Matrix01_01.jpg
http://www.bonesusvi.com/images/2006/Kelly%20and%20the%20cow.jpg

# Adding Strings

```
/* Just plain wrong */
proto_tree_add_text(tree, tvb, 0, 50, tvb_get_ptr(tvb, 0, 50));

/* Still bad */
proto_tree_add_text(tree, tvb, 0, 50, "%s", tvb_get_ptr(tvb, 0, 50));

/* Best */
proto_tree_add_text(tree, tvb, 0, 50, "%s", tvb_format_text(tvb, 0, 50));
proto_tree_add_item(tree, hf_my_ft_bytes_item, …);
```

# Columns

- epan/column-utils.h
- Accessed via pinfo
- Enums for each type (COL_PROTOCOL, COL_INFO)
- Don't use `check_col` any more

```
col_set_str(pinfo->cinfo, COL_PROTOCOL, "TCP");
col_set_str(pinfo->cinfo, COL_INFO, "[TCP segment of a reassembled PDU]");
```

# Memory management

- Manual: GLib
  - g_malloc(), g_free()
- Automatic: epan/emem.h
  - ep_alloc()
  - se_alloc()
  - Strings (static & growable)
  - Binary trees
  - Stacks
  - Fast!

# Let's Make a Dissector!

- Copy from README.developer or existing dissector
- Place in epan/dissectors (built-in)
- Add to DISSECTOR_SRC in Makefile.common
- More initial work for plugins

# Gopher Protocol

- RFC 1436
- Text-based (mostly)
- Variable-length fields

# Gopher Directory Entities

- Line-based
- Last line empty

| Field | Length |
|---|---|
| Type | 1 |
| Item name | 0 – 70 |
| ASCII Tab | 1 |
| Selector | 0 – 255 |
| ASCII Tab | 1 |
| Host | 0 – 255? |
| ASCII Tab | 1 |
| Port | 0 – 5 |
| ASCII CR-LF | 2 |

# Challenges

- "Simplicity is intentional"

  …so what does this packet contain?

- What are our minimum lengths?

# Example

Minimal Gopher Dissector

# Value Strings

- Map integers to strings
- `val_to_str()`, `match_strval()`
- Also range strings, T/F strings, string strings, and bitfields

```
static const value_string auth_vals[] = {
  {0,   "Authentication Request"},
  {1,   "Authentication Response"},
  {847, "Look! Ice cream man!"},
  {0,   NULL}
};
```

# Example

## Directory Dissection

# Dissector Deficiencies

- Primitive state tracking
- No reassembly
- Limited response support
- Wire data doesn't match spec (GASP!)

# Strings

- GLib internals
  - g_str*(), g_string_*();
- ep_str*() and tvb_get_str*()
- epan/strutil.h, epan/to_str.h
- ep_strbuf*()

# ep_strbuf Example

```
emem_strbuf_t *flags_strbuf = ep_strbuf_new_label("<None>");
const gchar *fstr[] = {"FIN", "SYN", "RST", "PSH", "ACK", "URG", "ECN", "CWR"};
gboolean first_flag = TRUE;


for (i = 0; i < 8; i++) {
  bpos = 1 << i;
  if (tcph->th_flags & bpos) {
    if (first_flag) {
      ep_strbuf_truncate(flags_strbuf, 0);
    }
    ep_strbuf_append_printf(flags_strbuf, "%s%s", first_flag ? "" : ", ",
                            fstr[i]);
    first_flag = FALSE;
  }
}
```

# Plugins

- Live in /plugins
- Separate DLL / shared object
- For each plugin:
  1. Load it
  2. Look for init routines
  3. Run them

# Plugin or Built-in?

| | Plugin | Built-in |
|---|---|---|
| Licensing | Proprietary? | GPLv2 |
| Complexity | Some | Minimal |
| Distribution | DLL | Application |

# Creating a Plugin

- Read doc/README.plugins
- Place in /plugins/xyzzy
  - Dissector source
  - Makefiles
  - Boilerplate
- plugin.c wrapper auto-generated

# Distributing Your Code

- Can you?
- Should you?

# Contributing Your Code

1. Fuzz! `tools/fuzz-test.sh`
2. Check! `tools/checkAPIs.pl`
3. Generate a patch

   ```
   svn diff > /tmp/skype.patch
   ```

4. Patch + sample capture → bugs.wireshark.org

# Fuzzing Example

```
cd wireshark-gtk2
../tools/fuzz-test.sh /tmp/*.pcap

../tools/fuzz-test.sh: line 56: ulimit: cpu time: cannot modify limit: Invalid argument
Running ./tshark with args: -nVxr (forever)

Starting pass 1:
  c:\cygwin\tmp\buildbot.test.pcap:  OK
Starting pass 2:
  c:\cygwin\tmp\buildbot.test.pcap:  OK
    …
```

# Common Mistakes

- Working too hard
- Not using `proto_tree_add_item`
- Extracting packet data yourself
- Assuming packets are processed in order
- Setting columns and offsets inside `if (tree)`
- `tvb_reported_length_remaining()`
- Creating a bunch of tvbuffs
- `abort` or `g_assert` in your dissector

# Protocol Preferences

- Uints, Bools, Enums, Strings, Ranges
- General registration
  - Protocol + Callback
- Preference registration
  - Name
  - Data pointer (usually global)
- Stored in main prefs file
- See also: UATs

# Preferences Example

```
static guint g_xyzzy_tcp_port = TCP_PORT_XYZZY;

proto_xyzzy = proto_register_protocol(…);

xyzzy_module = prefs_register_protocol(proto_xyzzy,
  proto_reg_handoff_xyzzy);

prefs_register_uint_preference(
  xyzzy_module, "tcp.port", "Xyzzy TCP Port",
  "TCP port for xyzzy messages", 10, &g_xyzzy_tcp_port);
```

# Example

Gopher Preferences

# Keeping State

- Order not guaranteed
  - pinfo->fd->flags.visited
- Within your dissector
  - Normal C variables
- Up & down the stack
  - pinfo->private_data
- Across calls
  - p_add_proto_data
  - Conversations

# Protocol Data Example

```
per_packet_info = p_get_proto_data(pinfo->fd, proto_vnc);

if(!per_packet_info) {
        per_packet_info = se_alloc(sizeof(vnc_packet_t));

        per_packet_info->state = per_conversation_info->vnc_next_state;
        per_packet_info->preferred_encoding = -1;

        p_add_proto_data(pinfo->fd, proto_vnc, per_packet_info);
}

/* Packet dissection follows */
switch(per_packet_info->state) {
```

# Conversations

- Packets between address:port pairs
- Versatile creation:

  `find_conversation` + `conversation_new`
- Easy creation:

  `find_or_create_conversation`
- Adding / getting data

  `conversation_add_proto_data`

  `conversation_get_proto_data`

# Conversation State Example

```
/*
 * Find or create the conversation for this.
 */
conversation = find_or_create_conversation(pinfo);

/*
 * Is there a request structure attached to this conversation?
 */
session_state = conversation_get_proto_data(conversation, proto_smtp);
if (!session_state) {
  /*
   * No - create one and attach it.
   */
  session_state = se_alloc(sizeof(struct smtp_session_state));
  session_state->smtp_state = SMTP_STATE_READING_CMDS;
  session_state->crlf_seen = FALSE;
  session_state->data_seen = FALSE;
  session_state->msg_read_len = 0;
  session_state->msg_tot_len = 0;
  session_state->msg_last = TRUE;
  session_state->last_nontls_frame = 0;

  conversation_add_proto_data(conversation, proto_smtp, session_state);
}
```

# TCP Reassembly

- TCP messages & tvbuffs have different boundaries
- `tcp_dissect_pdus()` to the rescue!
- epan/dissectors/packet-tcp.h
- What about other reassembly?

# Using tcp_dissect_pdus()

```
static void
dissect_dns_tcp_pdu(tvbuff_t *tvb, packet_info *pinfo, proto_tree *tree)
{
  col_set_str(pinfo->cinfo, COL_PROTOCOL, "DNS");

  dissect_dns_common(tvb, pinfo, tree, TRUE, FALSE, FALSE);
}

static void
dissect_dns_tcp(tvbuff_t *tvb, packet_info *pinfo, proto_tree *tree)
{
  tcp_dissect_pdus(tvb, pinfo, tree, dns_desegment, 2, get_dns_pdu_len,
      dissect_dns_tcp_pdu);
}
```

```
proto_reg_handoff_dns(void)
{
  dissector_handle_t dns_udp_handle;
  dissector_handle_t dns_tcp_handle;
```

```
  dissector_add("tcp.port", TCP_PORT_DNS, dns_tcp_handle);

}
```

# General Reassembly

- Collect fragments: `fragment_add_XXX`
- Create tvb: `tvb_new_XXX`
- Create detail tab: `add_new_data_source`
- Dissect the child data: `dissect_XXX`

# IP Defragmentation

```
/* If ip_defragment is on, this is a fragment, we have all the data
 * in the fragment, and the header checksum is valid, then just add
 * the fragment to the hashtable.
 */
save_fragmented = pinfo->fragmented;
if (ip_defragment && (iph->ip_off & (IP_MF|IP_OFFSET)) &&
    tvb_bytes_exist(tvb, offset, pinfo->iplen - pinfo->iphdrlen) &&
    ipsum == 0) {
      ipfd_head = fragment_add_check(tvb, offset, pinfo,
                  iph->ip_p ^ iph->ip_id ^ src32 ^ dst32,
                  ip_fragment_table,
                  ip_reassembled_table,
                  (iph->ip_off & IP_OFFSET)*8,
                  pinfo->iplen - pinfo->iphdrlen,
                  iph->ip_off & IP_MF);

  next_tvb = process_reassembled_data(tvb, offset, pinfo, "Reassembled IPv4",
    ipfd_head, &ip_frag_items, &update_col_info, ip_tree);
} else {
```

# Exceptions

- Automatic

```
offset = 234567890;
uid = tvb_get_ntohs(tvb, offset);
```

- Manual

```
THROW(ReportedBoundsError);
DISSECTOR_ASSERT(offset < 300);
REPORT_DISSECTOR_BUG("That wasn't cheese…");
```

# Error Reporting

- Bad:

```
g_assert(len <= MAX_LEN);
```

- Sort-of-OK:

```
fprintf(stderr, "Oops.");
proto_tree_add_debug_text(…);
```

- Better: Expert Info

# Expert Info

- Adds to expert windows
- Similar to syslog
- epan/expert.h, epan/expert.c

```
expert_add_info_format(pinfo, ti, PI_MALFORMED, PI_ERROR,
                       "Corrupted data segment");
expert_add_info_format(pinfo, ti, PI_SEQUENCE, PI_NOTE,
                       "Less horseradish next time");
```

# Portability Tips

- We run on Windows (32 & 64), Linux, Solaris, OS X, FreeBSD, NetBSD, OpenBSD, AIX, HP-UX, …
- GLib types
- Old compilers (Visual C++ 6.0)
  - No C++ comments
  - No C99?!

```
#include <msvc_rant.h>
```

# Portability tips 2

- No malloc, sprintf, strcpy, open…
- `sizeof` and `strlen` returns a `size_t`
- Use ep_ and se_ allocated memory
- `#ifdef _WIN32 /* not WIN32 */`

# How To Crash Wireshark

- Dereference a NULL pointer
- Over- (or under-) run a buffer
- Pass a NULL string to a printf-style function

  …sometimes
- Global pointer to ep_allocated memory

# Check Your Inputs

```
elem_desc_len = tvb_get_ntohs(…);

while (desc_bytecnt != 0) {
  elem_bytecnt = elem_desc_len;

  if (elem_bytecnt > desc_bytecnt)
    elem_bytecnt = desc_bytecnt;

  dissect_something_or_other(…);

  offset += elem_bytecnt;
  desc_bytecnt -= elem_bytecnt;
}
```

# Speaking of Loops…

```
guint8 pdu_len, el_len;
int offset;
pdu_len = tvb_get_guint8(tvb, offset);
offset++
while (pdu_len > 0) {
        el_len = tvb_get_guint8(tvb, offset);
        dissect_our_pdu(…);
        offset += el_len;
        pdu_len -= el_len;
}
```

# What's the Difference?

```
some_string = tvb_get_string(tvb, 0, 20);

col_add_fstr(pinfo->cinfo, COL_INFO, some_string);

col_set_str(pinfo->cinfo, COL_INFO, some_string);
```

# Making Your Own Package

- Why?
- doc/README.packaging
- version.conf + make-version.pl

# Disk Requirements

| | |
|---|---|
| • Sources (plain) | 430 MB |
| • Sources (compiled) | 850 MB |
| • Support libs | 250 MB |
| • Cygwin | .5 – 2.0 GB |
| • Python | 50 MB |

# Why won't you add my code?

- Is it well-written?
- Did you fuzz it?
- Did you send along a capture file?
- Should you ping someone?

# Ptvcursors

- Protocol Tree TVBuff Cursor
- Easy way to add a bunch of static items

# Ptvcursor Example

```
ptvcursor_t *cursor;
int offset = 0;

cusor = ptvcursor_new(tree, tvb, offset);
ptvcursor_add(cursor, hf_stream_addr, 1,
        FALSE);
        /* more ptvcursor_add calls */
ptvcursor_add(cursor, hf_salmon_count, 4,
        FALSE);
offset = ptvcursor_current_offset(cursor);
ptvcursor_free(cursor);
return offset;
```

# Automatic Generation

- ASN.1
- CORBA IDL
- Samba PIDL
- Protomatics

# Further Information

- http://anonsvn.wireshark.org/wireshark/trunk
- http://www.wireshark.org/develop.html
- Wireshark Developer's Guide
- doc/README.developer
- wireshark-dev@wireshark.org