Wireshark Dissectors and Plugins

June 16, 2009

Gerald Combs

Lead Developer | Wireshark

SHARKFEST **'09**

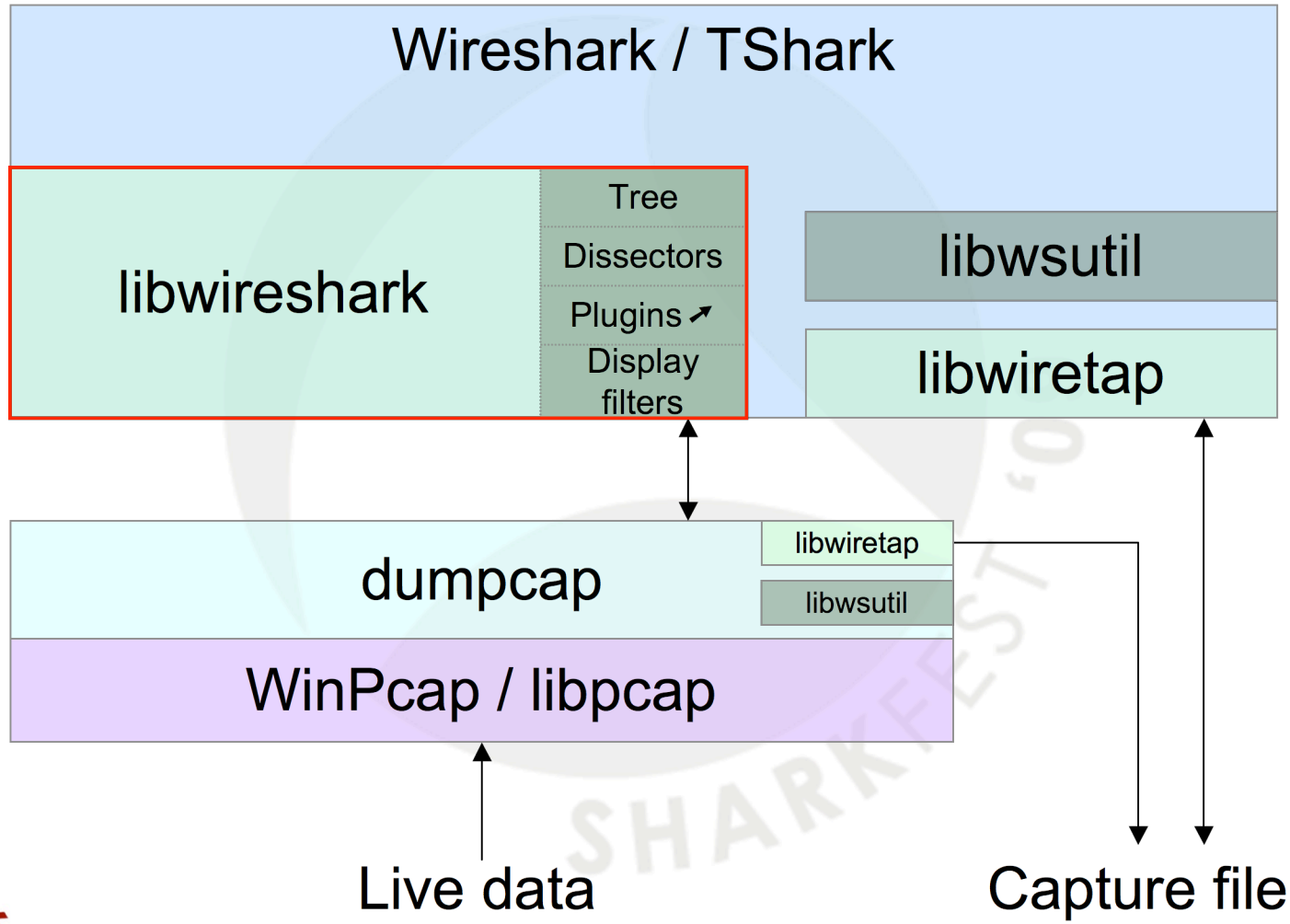
Stanford University

June 15-18, 2009

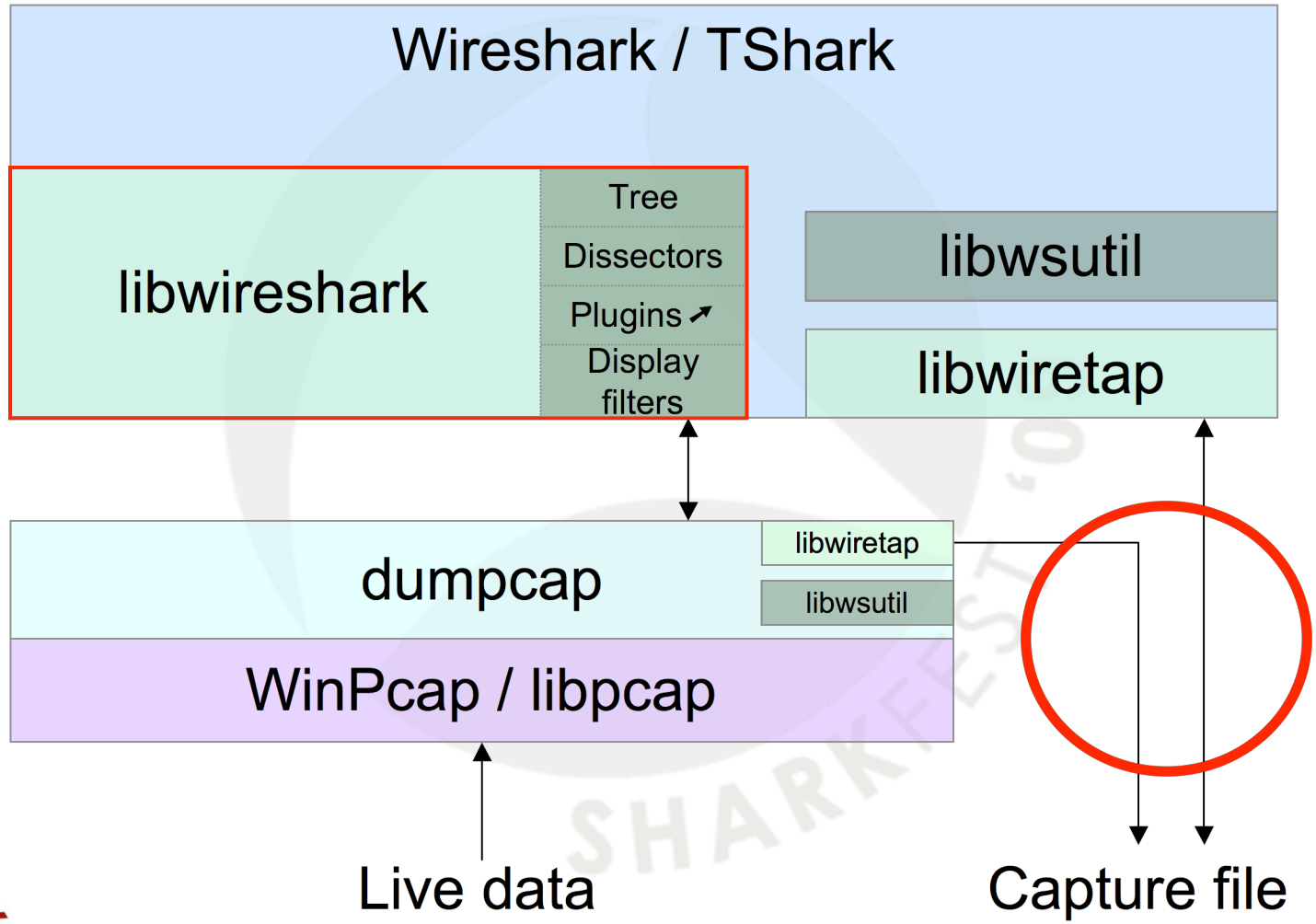
Overview

- Distributed development
- Plain old boring C
- Multi-platform
- Multi-interface
- Open Source (GPL)
- Rapid development pace

Application Architecture



Application Architecture



Build Requirements

- Source code: (like, duh.)
- Compilers: Visual C++ or gcc
- Libraries: WinPcap/libpcap, GLib, GTK+, zlib, libsmi, …
- Support tools: Python, Perl, Linux/UNIX shell

Build Requirements - Windows

- Visual C++ 6.0 to **2008**
- Cygwin
- Python 2.4+ (Not 3.0)
- NSIS, TortoiseSVN

Disk Requirements

• Sources (plain)	250 MB
• Sources (compiled)	700 MB
• Support libs	250 MB
• Cygwin	650 MB
• Python	50 MB
• Total (compiled)	1.65 GB

Getting the Code

- Subversion

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

  http://anonsvn.wireshark.org/wireshark/trunk-1.2

- Tar files

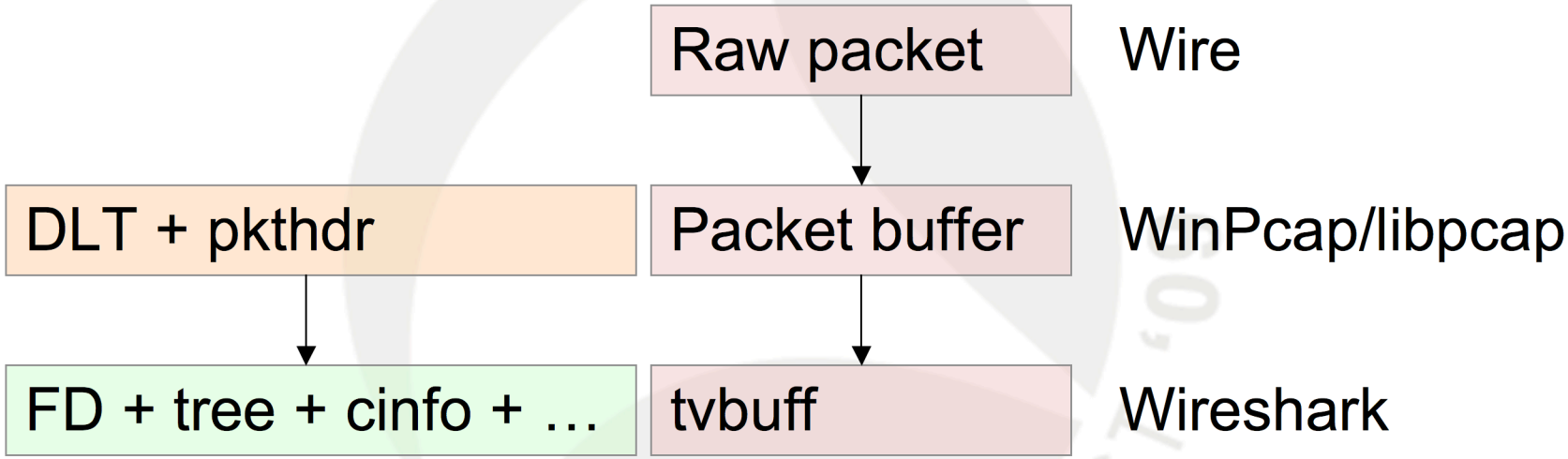
  http://www.wireshark.org/download/src

- Want to send us a patch? Use SVN!

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



Parts of a Dissector

- Registration routines
- Globals (value strings)
- Hfinfo structs
- Dissection routines

Registration

- Initialization: proto_register_xyzzy();
  - Preference callbacks
- Handoffs: proto_reg_handoff_xyzzy();
- make-dissector-reg.py → epan/dissectors/register.c
- Two passes

Core Data Structures

- Provided to you:
  - tvbuff: Protocol data access
  - packet_info, frame_data: Packet meta-info
  - proto_tree: Detail tree
- You provide:
  - header_field_info

Getting Called

- Normal
  - dissector_add
- 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()
```

tv_buff: Testy, virtual buffers

- Data buffers & extraction
- epan/tvbuff.h
- tvb_get_XXX(tvb, offset, …);
- Make your own! Impress your friends!
- Safe (mostly) except for tvb_get_ptr()

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

- Protocol atoms
- Data type, filter name, descriptions
- Enums - Value/Range/TF Strings
- epan/proto.h
- Unique

proto_tree

- Detail tree (middle pane)
- Might be NULL
- epan/proto.h
- proto_tree_add_XXX(tree, hf_index, tvb, offset, length, …);
- proto_tree_add_item
- Can be hidden or “generated”
- Avoid proto_tree_add_text()

Adding Tree Items

```
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, …);
```

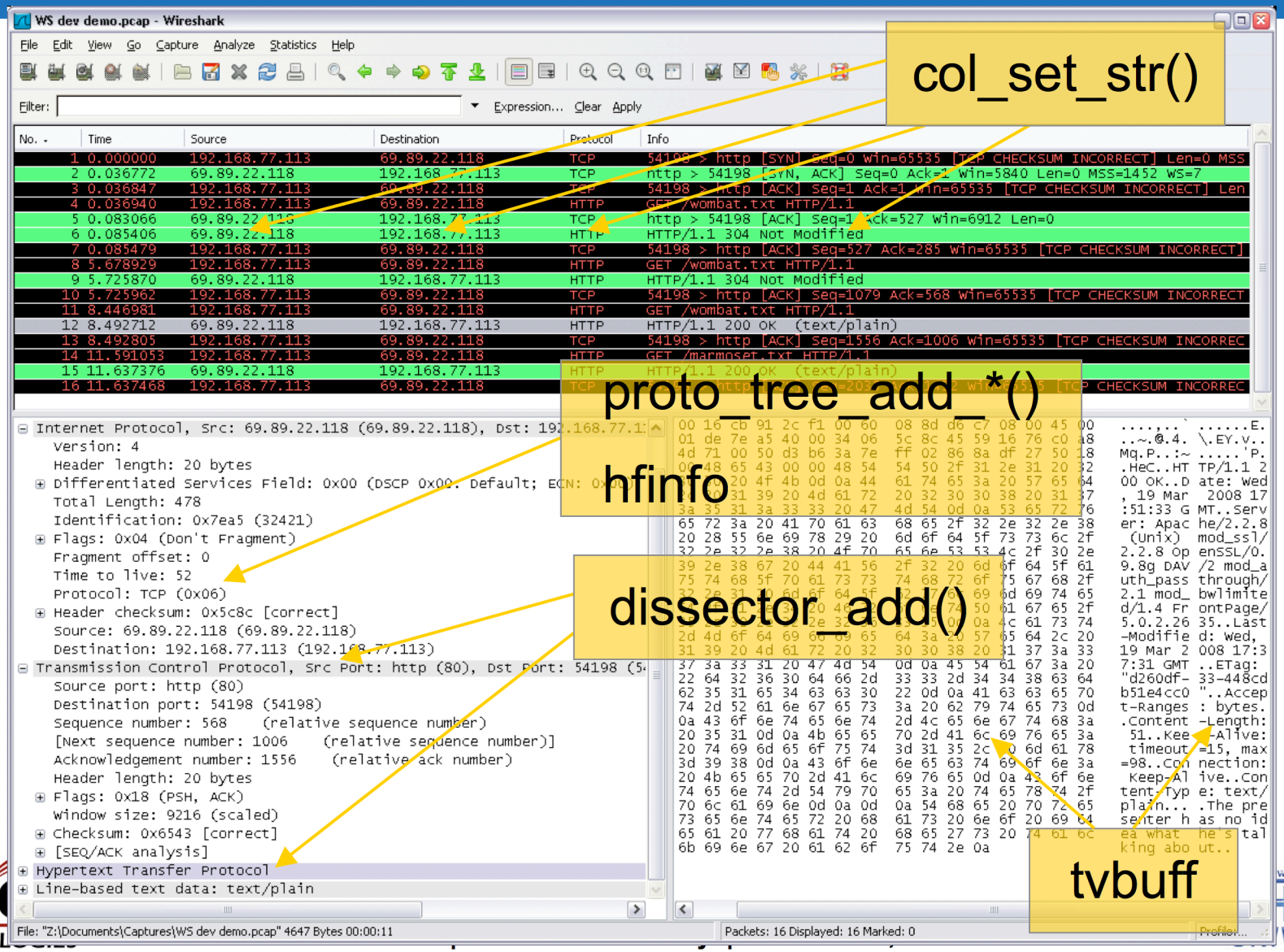
Adding Raw Data

```
/* 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(…); /* FT_BYTES */
```

Columns

- epan/column-utils.h
- Accessed via pinfo
- Enums for each type (COL_PROTOCOL, COL_INFO)
  - col_set_str(pinfo->cinfo, COL_PROTOCOL, "TCP");
  - col_add_fstr

UI Element Origins



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 it to Makefile.common (CLEAN_DISSECTOR_SRC)
- More initial work for plugins

Banana Protocol

- Used by the Twisted Python framework
- Serialized data structures (s-expressions)
- Variable-length fields
- …but not TLVs

Banana Elements

ID	Type
0x80	List
0x81	Integer
0x82	String
0x83	Negative Integer
0x84	Float
0x85	Large Integer
0x86	Large Negative Integer
0x87	Perspective Broker Dictionary

Challenges

- Overloaded value/length
- Unbounded lengths

Example

Minimal Banana 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

Banana Tree

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

Banana Prefs

Plugins

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

Plugin or Built-in?

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

We'll focus on built-in

Distributing Your Code

- Can you?
- Should you?

Creating a Plugin

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

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

```
packet_state = p_get_proto_data(pinfo->fd, proto_eap);
if (packet_state == NULL) {
    packet_state = se_alloc(sizeof (frame_state_t));
    packet_state->info = eap_reass_cookie;
    p_add_proto_data(pinfo->fd, proto_eap, packet_state);
```

Conversation Example

```
conv = find_conversation(pinfo->fd->num, &pinfo->src, &pinfo->dst, pinfo->ptype, pinfo->srcport, pinfo->destport, 0);

 if (NULL == conv) {
     conv = conversation_new(pinfo->fd->num, &pinfo->src, &pinfo->dst, pinfo->ptype, pinfo->srcport, pinfo->destport, 0);
     conversation_set_dissector(conv, fix_handle);
 }
```

Conversation State Example

```
conversation_state =
   conversation_get_proto_data(conversation, proto_eap);
 if (conversation_state == NULL) {
   /* Attach state information to the conversation. */
   conversation_state = se_alloc(sizeof (conv_state_t));
   conversation_state->eap_tls_seq = -1;
   conversation_state->eap_reass_cookie = 0;
   conversation_state->leap_state = -1;
   conversation_add_proto_data(conversation, proto_eap,
   conversation_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()

```
#define MIN_MSG_LEN 4
static void dissect_xyzzy(…) {
    tcp_dissect_pdus(tvb, pinfo, tree, TRUE,
    MIN_MSG_LEN, get_xyzzy_message_len,
    dissect_xyzzy_message);
}
static void dissect_xyzzy_message(…) {
    /* Actual dissection */
}
static guint get_xyzzy_message_len(…) {
    /* Return message length */
}
```

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

Exceptions

- Automatic

  offset = 234567890;

  uid = tvb_get_ntohs(tvb, offset);

- Manual

  THROW(ReportedBoundsError);

  DISSECTOR_ASSERT(offset < 300);

  REPORT_DISSECTOR_BUG("Cheese smells funny");

- Expert

  epan/except.h, epan/proto.h

Expert Info

```
expert_add_info_format(pinfo, ti, PI_MALFORMED,
    PI_ERROR, "Corrupted data segment");
expert_add_info_format(pinfo, ti, PI_SEQUENCE,
    PI_NOTE, "Try socks THEN shoes next time");
```

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

Example

Banana Expert

Dissector Deficiencies

- No type or value/length fields
- Highlights are funny
- Zero-padded length/values
- Wire data doesn't match spec (GASP!)
- …and neither do we
- Many types untested
- Not enabled by default

Portability Tips

- We run on Windows (32 & 64), Linux, Solaris, OS X, FreeBSD, NetBSD, OpenBSD, AIX, HP-UX, …
- GLib types, not C99 (e.g. guint8 instead of uint8_t)
- No C++ comments

Portability tips 2

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

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?

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

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

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

Other Ways to Crash

- Pass a NULL string to a printf-style function
- Global pointer to ep_allocated memory

Contributing Your Code

1. Fuzz!
2. Run check scripts (checkAPIs.pl)
3. Generate a patch

   svn diff > ~/Desktop/banana.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
    …
```

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
- Next session

Bonus Material

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;
```

Strings

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

Making Your Own Package

- Why?
- doc/README.packaging
- version.conf
- NSIS

Automatic Generation

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