Introduzione alla prorammazione di rete Libnet e Libpcap

Click, embyte, snifth
Hackmeeting 2003

Torino, 20 giugno 2003

TCP/IP Suite & Layering

Introduzione

- Definizione di protocollo
  possibilia' di intercomunicazione
- Perche' il TCP/IP
- Accenni storici
  1960 progetto finizanziato dal governo
  1990 networking tra computer
- Concetto di porta e servizio
- Pacchetti e datagrammi

TCP/IP Suite & Layering

IP: Internet Protocol [rfc 791]

- Datagram

  header (20 bytes unless option)

  area dati

- Unreliable

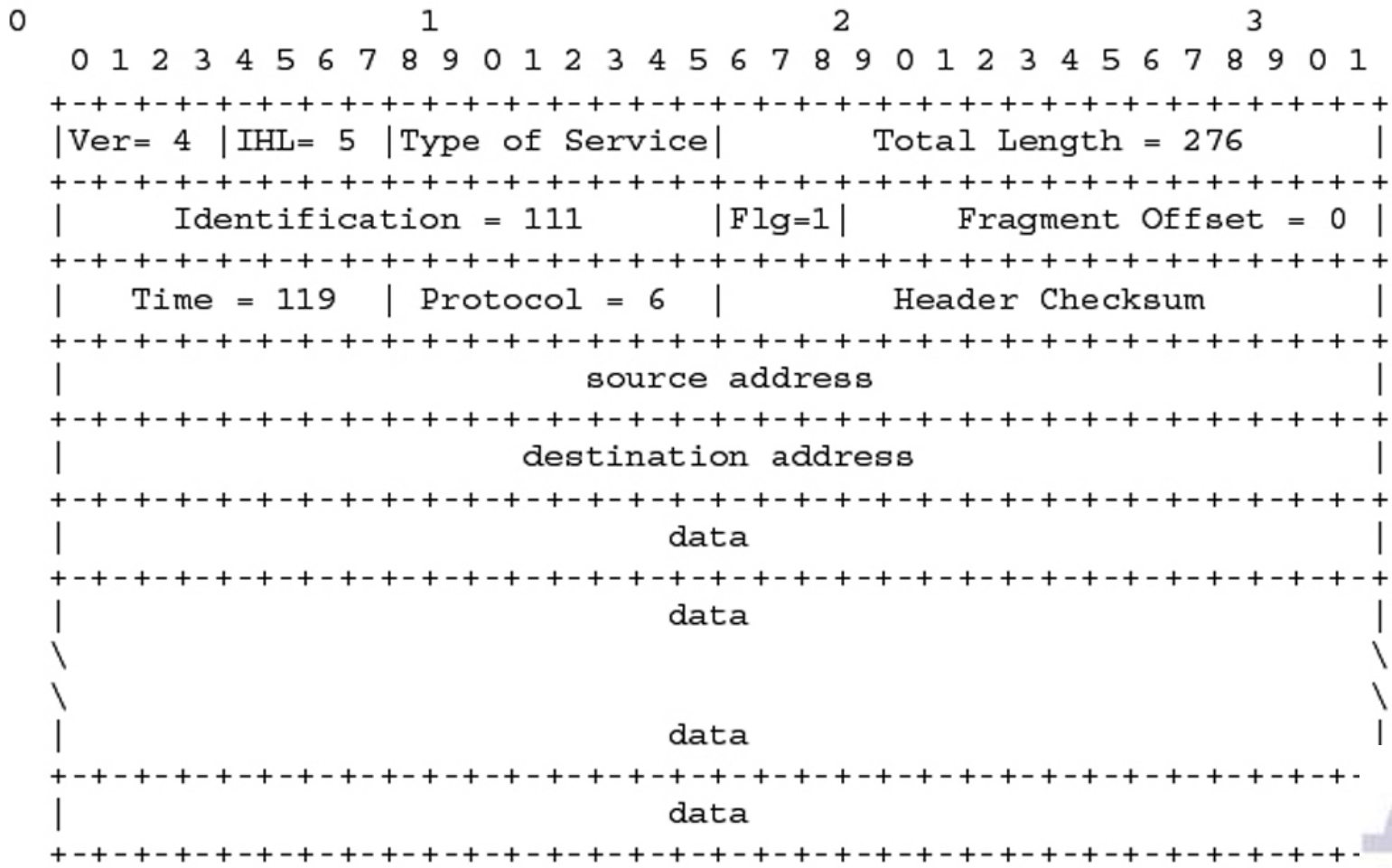
- ICMP

- Connectionless

  A-B non sempre nello stesso ordine

TCP/IP Suite & Layering

- **Header Ip**



TCP/IP Suite & Layering

ICMP : Internet Control Message Protocol [rfc 792]

- Protocollo di basso livello

  Network layer in unione con il protocollo IP

- Trasporto dei messaggi di errore della rete

- Trasporto dei messaggi di richiesta

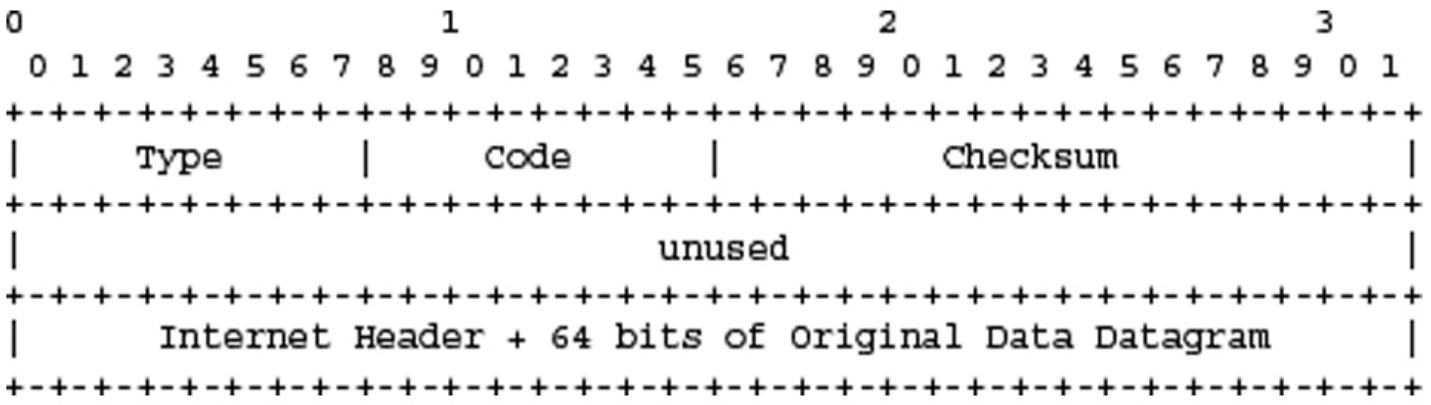
  Richieste di informazioni e risposte

- Type 0 risposta echo

  Type 3 destinazione irraggiungibile

TCP/IP Suite & Layering

- **ICMP message**



Es: Type 0 risposta echo

Type 3 destinazione irraggiungibile

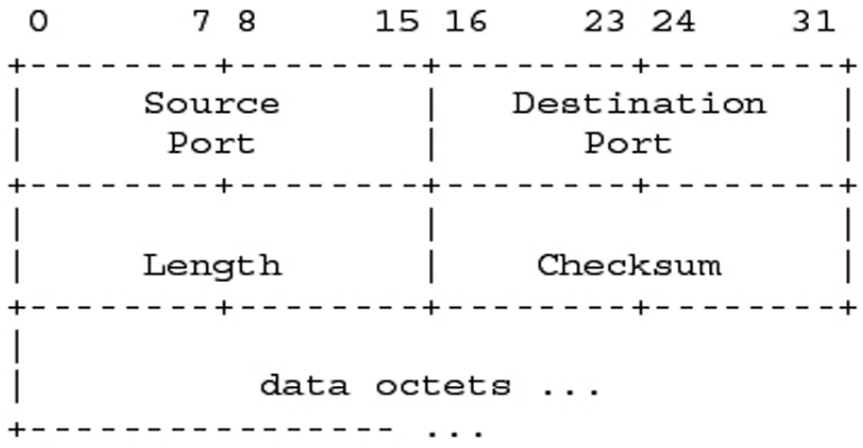
TCP/IP Suite & Layering

UDP : User Datagram Protocol [rfc 768]

- Consegna dei dati veloce ma non garantita
- Datagram Oriented
- Scarto silenzioso dei pacchetti senza generazione di errore (checksum)
- Connectionless
- Meno affidabile ma piu' veloce

TCP/IP Suite & Layering

- **Header udp**



TCP/IP Suite & Layering

- **TCP : Transfer Control Protocol [rfc 793]**

- Connessione full-duplex

  Trasferimento bidirezionale in cui si ha una reciproca conferma di ricezione dei pacchetti

- 1-1023 well-known port

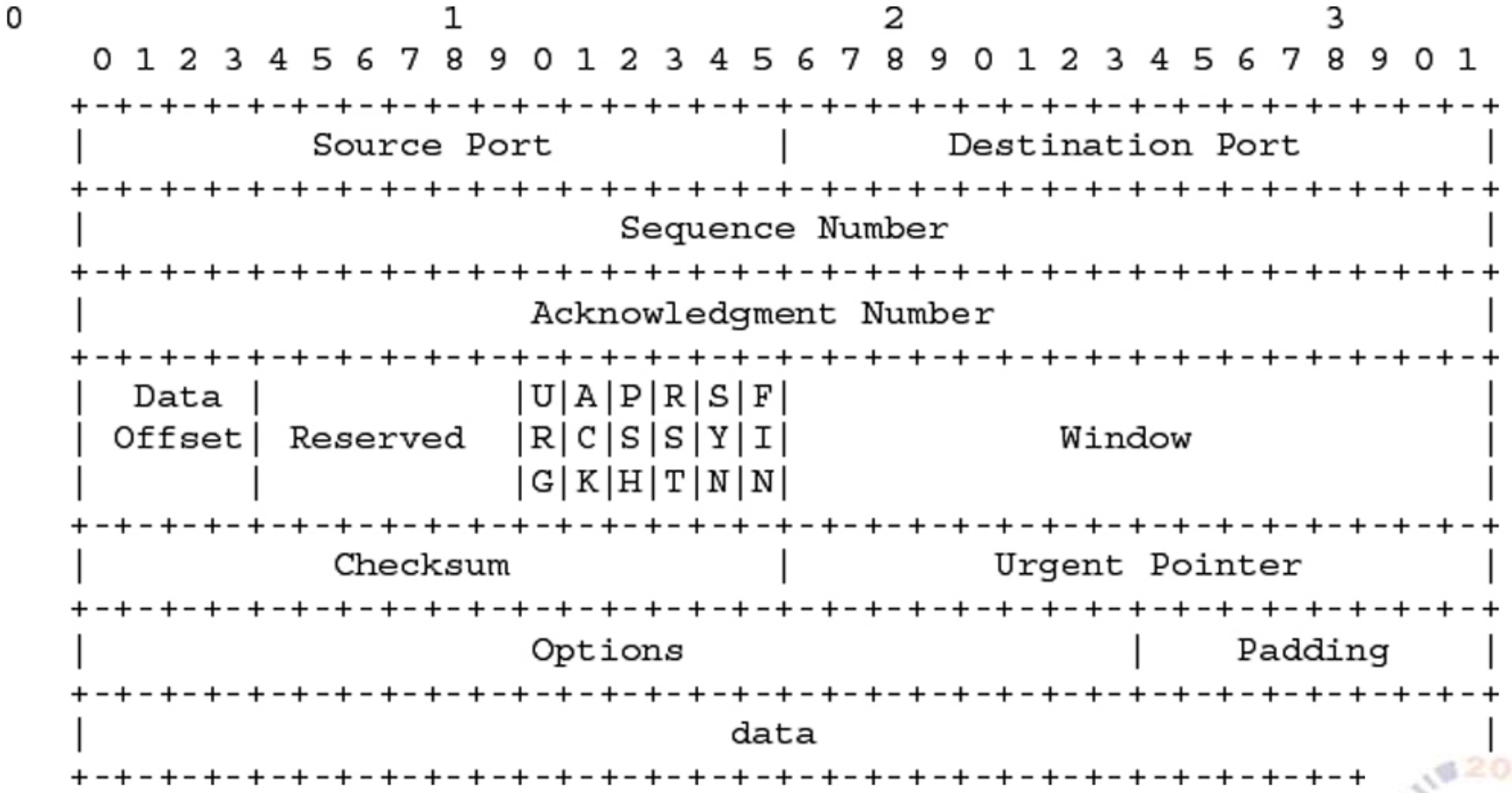
  Solo da root posso aprire queste porte

- Protocollo “affidabile”

- Handshaking a tre vie

TCP/IP Suite & Layering

- **Header TCP**



TCP/IP Suite & Layering

ARP : Adress Resolution Protocol

- Risoluzione di un indirizzo IP nel relativo indirizzo MAC

- Media Access Control indirizzo fisico dell'host

a) Verifica se il server fa parte della stessa rete

b) L'indirizzo IP del server viene confrontato con la subnet mask del client

c) se il server appartiene alla stessa rete:

1) il client prima controlla se l'indirizzo del server e' nella propria ARP cache

in caso negatio viene inviato un pacchetto con la richiesta di IP e relativo MAC che andranno nella cache ARP del client

- Gli host a cui non e' diretto il pacchetto lo ignoreranno dopo averlo esaminato

TCP/IP Suite & Layering

Lo stack TCP/IP

Application	Comuni applicazioni utente	FTP,TELNET,SMTP, POP3,HTTP,DNS
Transport	Trasporto dei pacchetti end-to-end	TCP e UDP
Network	Logical Adressing e routing	IP, ICMP, IGMP, RIP,OSPF, DHCP
Data Link	Interfaccia con i medium fisici	ARP,RARP,SLIP/PPP

Packet Capture Library

- **Introduzione alla pcap sotto linux**

  Storia della libreria
  Dove utlizzarla
  Vantaggi di portabilita'

- **Installazione**

  www.tcpdump.org
  libpcap-0.7.2.tar.gz
  bishop@mystaya:~/libpcap-0.7.2/ ./configure
  root@mistaya:~/libpcap-0.7.2/ make && make install

Packet Capture Library

- **Principali funzioni**

- pcap_t *pcap_open_live(char *device, int snaplen, int promisc,
int to_ms, char *errbuf)

```
#include<pcap.h>
        (...)
        pcap_t *descr;
        char errbuf[PCAP_ERRBUF_SIZE];
        descr = pcap_open_live(dev, BUFSIZ, 1, 0, errbuf);
        if(descr==NULL) printf("Error: %s\n",errbuf);
        (...)
```

- pcap_t *pcap_open_offline(char *fname, char *errbuf)

```
#include<pcap.h>
        (...)
        pcap_t *descr;
        char *filename;
        char errbuf[PCAP_ERRBUF_SIZE];
        descr = pcap_open_offline(filename, errbuf);
        if(descr==NULL) printf("Error: %s\n",errbuf);
        (...)
```

Packet Capture Library

pcap_dumper_t *pcap_dump_open(pcap_t *p, char *fname)

```
#include<pcap.h>
        (...)
        pcap_t *descr;
        pcap_dumper_t *dumper;
        char *filename;
        char errbuf[PCAP_ERRBUF_SIZE];
        descr = pcap_open_live(eth0, BUFSIZ, 1, 0, errbuf);
        dumper = pcap_dump_open(descr, filename);
        if(dumper==NULL) printf("Error\n);
        (...)
```

int pcap_lookupnet(char *device, bpf_u_int32 *netp,
bpf_u_int32 *maskp, char *errbuf)

```
        #include<pcap.h>
        (...)
        bpf_u_int32 maskp;
        bpf_u_int32 netp;
        if(pcap_lookupnet(dev,&netp,&maskp,errbuf)==-1)
              printf("Error: %s\n",errbuf);
```

Packet Capture Library

int pcap_compile(pcap_t *p, struct bpf_program *fp, char *str,
int optimize, bpf_u_int32 netmask)

```
#include<pcap.h>
(...)
struct bpf_program fp;

(ricaviamo il descr con la pcap_open_live() o la
pcap_open_offline() )

if(pcap_compile(descr,&fp,"src 192.168.1.3",0,maskp)==-1)
        printf("Error: %s\n",pcap_geterr(descr));
```

int pcap_setfilter(pcap_t *p, struct bpf_program *fp)

```
if(pcap_setfilter(descr,&fp) == -1)
printf("Error calling pcap_setfilter\n");
```

Packet Capture Library

u_char *pcap_next(pcap_t *p, struct pcap_pkthdr *h)

```
u_char *packet;
struct pcap_pkthdr hdr;
packet = (u_char *) pcap_next (descr, &hdr))
```

void pcap_freecode(struct bpf_program *);

```
pcap_freecode(&fp);
```

void pcap_dump(u_char *user, struct pcap_pkthdr *h,
u_char *sp)

```
pcap_dump((u_char *)dumper,&hdr,packet);
```

Libnet-Packet Shaping Library

LIBNET 1.1 (c) 1998 - 2002 Mike D. Schiffman <mike@infonexus.com>

Breve introduzione alla libreria LIBNET 1.1 sotto Linux

Cos'è?
Perchè utilizzarla con la libpcap?

Storia della libreria

0.1	01.05.1998
1.0.2a	02.06.2001
1.1.0	**08.05.2002**

Vantaggi di portabilita'

BSD/OS 4.x
Cygwin
OS/X
FreeBSD 4.x, 5.x
OpenBSD 2.x, 3.x
HPUX 11.0
Linux 2.0.x, 2.2.x, 2.4.x
Solaris 2.x, 7, 8

Protocolli supportati

802.1q 802.2 802.3 ARP RARP CDP DATA DHCP BOOTP DNS Ethernet
ICMP IGMP IP IPSEC ISL NTP OSPF RIP STP TCP UDP VRRP

Libnet-Packet Shaping Library

Internal routine

- Memory allocation for packet headers
- Checksums (libnet_toggle_checksum() do disable)
- Packet header ordering
- Additional sanity checking

Installazione

Homepage : www.packetfactory.net/libnet
bishop@mystaya:~/Libnet-lastest/ ./configure
root@mistaya:~/Libnet-lastest/ make && make install

Note di compilazione

Ex: gcc -g -Wall -O2 `libnet-config --defines` SORGENTE.c -o BINARIO -lnet
("#include libnet.h")

-lnet linka la libreria oggetto libnet.a

$ libnet-config –defines
-D_BSD_SOURCE -D__BSD_SOURCE -D__FAVOR_BSD -DHAVE_NET_ETHERNET_H

NB: il comando libnet-config dal libnet 1.1 è deprecated per cui make install non lo copierà in $PREFIX/bin

Libnet-Packet Shaping Library

- **Funzioni principali**

```
libnet_t * libnet_init(int injection_type, char *device, char *err_buf);  /* Initialize context handle */

int  libnet_write(libnet_t *l);

void libnet_destroy(libnet_t *l);
```

- **libnet_init() details**

Constant	*Meaning*
LIBNET_LINK	Link-layer interface
LIBNET_RAW4	Raw sockets using Ipv4
LIBNET_RAW6	Raw sockets using IPv6
LIBNET_LINK_ADV	Link-layer interface (advanced)
LIBNET_RAW4_ADV	Raw sockets using IPv4 (advanced)
LIBNET_RAW6_ADV	Raw sockets using IPv6 (advanced)

Libnet-Packet Shaping Library

Miscellaneous

char * libnet_geterror(libnet_t *l);
void libnet_stats(libnet_t *l, struct libnet_stats *ls);
int libnet_getfd(libnet_t *l);

Address resolution functions

u_long libnet_name2addr4(libnet_t *l, u_char *host_name, u_short use_name);
u_char * libnet_addr2name4(u_long in, u_short use_name);

use_name could be: LIBNET_DONT_RESOLVE or LIBNET_RESOLVE

Randomize functions

int libnet_seed_prand(libnet_t *l);
u_long libnet_get_prand(int type);

Ex:

```
void randomize()
{
  libnet_seed_prand(l);
  id = (u_short) libnet_get_prand(LIBNET_PRu16);
  seq = libnet_get_prand(LIBNET_PRu32);
  ack = libnet_get_prand(LIBNET_PRu32);
  urgp = (u_short) libnet_get_prand(LIBNET_PRu16);
}
```

Libnet-Packet Shaping Library

Packet Creation

```
libnet_ptag_t          /* packet id on success, -1 on failure */
libnet_build_arp(
    u_short,           /* hardware address type */
    u_short,           /* protocol address type */
    u_char,            /* hardware address length */
    u_char,            /* protocol address length */
    u_short,           /* ARP operation type */
    u_char *,          /* sender hardware address */
    u_char *,          /* sender protocol address */
    u_char *,          /* target hardware address */
    u_char *,          /* target protocol address */
    u_char *,          /* payload or NULL if none */
    u_long,            /* payload length */
    libnet_t *,        /* libnet context pointer */
    libnet_ptag_t      /* packet id */ );

libnet_ptag_t libnet_build_ethernet(
    u_char *,          /* pointer to a 6 byte ethernet address */
    u_char *,          /* pointer to a 6 byte ethernet address */
    u_short,           /* type */
    u_char *,          /* payload (or NULL) */
    u_long,            /* payload length */
    libnet_t *,        /* libnet context pointer */
    libnet_ptag_t      /* packet id */ );
```

Libnet-Packet Shaping Library

```
libnet_ptag_t  libnet_build_ipv4(
    u_short,        /* entire packet length */
    u_char,         /* tos */
    u_short,        /* ID */
    u_short,        /* fragmentation flags and offset */
    u_char,         /* TTL */
    u_char,         /* protocol */
    u_short,        /* checksum */
    u_long,         /* source address */
    u_long,         /* destination address */
    u_char *,       /* pointer to packet data (or NULL) */
    u_long,         /* payload length */
    libnet_t *,     /* libnet context pointer */
    libnet_ptag_t   /* packet id */ );

libnet_ptag_t  libnet_build_icmpv4_echo(
    u_char,         /* type */
    u_char,         /* code */
    u_short,        /* checksum */
    u_short,        /* id */
    u_short,        /* sequence number */
    u_char *,       /* pointer to packet data (or NULL) */
    u_long,         /* payload length */
    libnet_t *,     /* libnet context pointer */
    libnet_ptag_t   /* packet id */ );
```

Libnet-Packet Shaping Library

```
libnet_ptag_t libnet_build_tcp(
    u_short,        /* Source port */
    u_short,        /* Destination port */
    u_long,         /* Sequence Number */
    u_long,         /* Acknowledgement Number */
    u_char,         /* Control bits */
    u_short,        /* Advertised Window Size */
    u_short,        /* Checksum */
    u_short,        /* Urgent Pointer */
    u_short,        /* length of payload if a protocol header - not data */
    u_char *,       /* Pointer to packet data (or NULL) */
    u_long,         /* payload length */
    libnet_t *,     /* libnet context pointer */
    libnet_ptag_t   /* packet id */ );

libnet_ptag_t libnet_build_udp(
    u_short,        /* source port */
    u_short,        /* destination port */
    u_short,        /* total length (header + data) */
    u_short,        /* checksum */
    u_char *,       /* pointer to packet data (or NULL) */
    u_long,         /* payload length */
    libnet_t *,     /* libnet context pointer */
    libnet_ptag_t   /* packet id */ );
```

References

I tre esempi mostrati durante il workshop saranno pubblicati insieme alle slides su:

www.spine-group.org

Libnet && Libpcap

www.packetfactory.net/libnet
www.tcpdump.org

TCP/IP Illustrated vol 1 Stevens