§.网络序与主机序

1.例子

```
#include <stdio.h>
int main()
{    int k = 0x12345678;
     unsigned char *p = (unsigned char *)&k;

     printf("0x%08x\n", k);
     printf("0=%d\n", *p);
     printf("1=%d\n", *(p+1));
     printf("2=%d\n", *(p+2));
     printf("3=%d\n", *(p+3));

     return 0;
}
```

1、X86 Windows
2、X86 Linux
3、Arm Linux

	X86系统	ARM系统
printf("0x%08x\n", k);	0x12345678	0x12345678
printf("0=%d\n", *p);	0=120	0=18
printf("1=%d\n", *(p+1));	1=86	1=52
printf("2=%d\n", *(p+2));	2=52	2=86
printf("3=%d\n", *(p+3));	3=18	3=120

0x12345678 =	00010002	00110100	01010100	01111000

低位在前存放			高位在前存放	
2000	0111 1000	*p	2000	0001 0010
2001	0101 0100	*(p+1)	2001	0011 0100
2002	0011 0100	*(p+2)	2002	0101 0100
2003	0001 0010	*(p+3)	2003	0111 1000

§.网络序与主机序

2.主机序

不同的CPU(和操作系统无关)有不同的字节序类型，这些字节序是指整数在内存中保存的顺序，称为主机序，常见的有两种：

★ Little endian：将低序字节存储在起始地址，地址低位存储值的低位，地址高位存储值的高位（小头序/小字序）

★ Big endian：将高序字节存储在起始地址，地址低位存储值的高位，地址高位存储值的低位（大头序/大字序）

=> 在网络中不同字序的设备间传输数据时，引发误解

§.网络序与主机序

3.网络序

网络字节顺序是TCP/IP中规定好的一种数据表示格式，它与具体的CPU类型、操作系统等无关，从而可以保证数据在不同主机之间传输时能够被正确解释。网络字节顺序采用Big endian排序方式

★ TCP/IP库函数提供的4个转换函数

u_short htons(u_short) : host to net(short) 把2字节长度的数据从主机序转换到网络序

u_short ntohs(u_short) : net to host(short) 把2字节长度的数据从网络序转换到主机序

u_long htonl(u_long) : host to net(long) 把4字节长度的数据从主机序转换到网络序

u_long ntohl(u_long) : net to host(long) 把4字节长度的数据从网络序转换到主机序

- 网络一般以unsigend形式考虑，实际可以signed
- 按二进制方式理解，以字节为单位转换
- 在64位系统下，long为8字节，htonl/ntohl仍然转换4字节

```
#include <stdio.h>
#include <arpa/inet.h>
int main()
{   long x1 = 0x87654321;
    printf("%d %d\n", x1, htonl(x1));
    printf("0x%x 0x%x\n", x1, htonl(x1));

    short x2 = 0x4321;
    printf("%d %d\n", x2, htons(x2));
    printf("0x%x 0x%x\n", x2, htons(x2));

    return 0;
}
```

X86 Linux		ARM Linux	
-2023406815	558065031	-2023406815	-2023406815
0x87654321	0x21436587	0x87654321	0x87654321
17185	8515	17185	17185
0x4321	0x2143	0x4321	0x4321

§.网络序与主机序

问题：如何用最快的方法实现转换htonl等转换?

★ Little endian

```
#define  htons(x)      ((((x)>>8) & 0x00ff)+(((x)&0x00ff)<<8))
#define  htonl(x)      (((x)>>24)&0x000000ff+(((x)&0x00ff0000)>>8)+
                                (((x)&0x0000ff00)<<8)+(((x)&0x000000ff)<<24))
```

★ Big endian

```
#define  htons(x)   (x)
#define  htonl(x)   (x)
```

★ ntohl/ntohs依次类推