

# **Implémentation d'un protocole de transfert sans pertes**



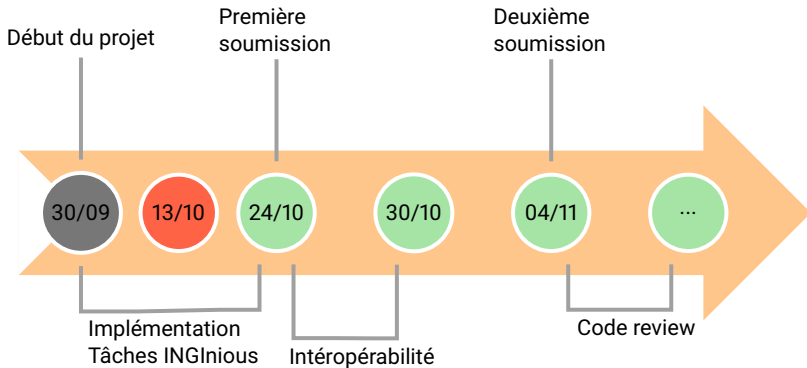
LINGI1341 – Projet de groupe  
October 13, 2016

Projet par groupe de deux étudiants

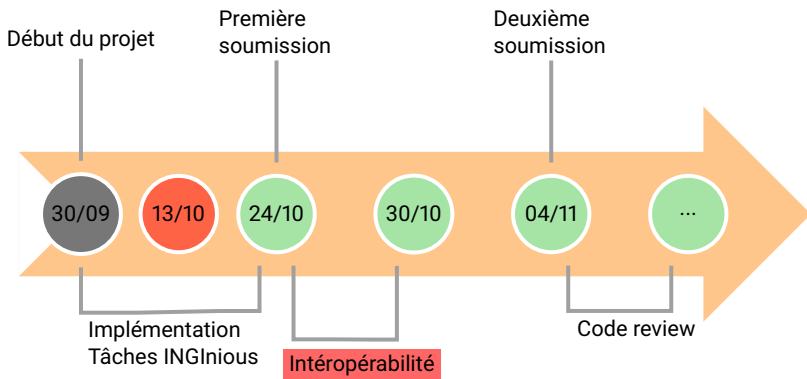
Projet par groupe de deux étudiants

cfr. forum Moodle : “Recherche de binôme”

# Planning



# Choix de la date pour les tests d'interopérabilité



# Implémentation d'un protocole de transfert sans pertes



- Organisation du projet
- Feedback INGINious

`char*` n'est pas synonyme de `string`!!!

`char*` n'est pas synonyme de `string`!!!

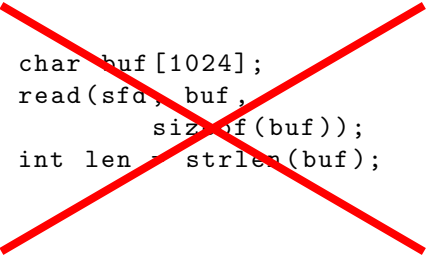
N'utilisez **JAMAIS** `strlen()`, `strcat()`, ...



## Vu dans vos codes...

```
char buf[1024];  
read(sfd, buf,  
      sizeof(buf));  
int len = strlen(buf);
```

## Tous les appels systèmes donnent la taille lue



```
char buf[1024];  
read(sfd, buf,  
      sizeof(buf));  
int len = strlen(buf);
```

```
char buf[1024];  
size_t len = read(sfd,  
                  buf, sizeof(buf));
```

Pas de garanties que `buf` contient un string valable  
(i.e., termine par un octet nul, et n'en contient aucun).

## Encoder une structure de donnée

```
unsigned int header = 1;
header = (header<<1);
header = header+1;
header = (header<<23);
header = header+LENGTH;
header = htonl(header);

temp = header;
for(i=0;i<4;i++){
    fprintf(f,"%c",(char)temp);
    temp = temp>>8;
}
fprintf(f,"0");
```

## Encoder une structure de donnée

```
unsigned int header = 1;
header = (header<<1);
header = header+1;
header = (header<<23);
header = header+LENGTH;
header = htonl(header);

temp = header;
for(i=0;i<4;i++){
    fprintf(f,"%c",(char)temp);
    temp = temp>>8;
}
fprintf(f,"0");
```

## Encoder une structure de donnée

```
uint32_t header = 1;
header = (header<<1);
header = header+1;
header = (header<<23);
header = header+LENGTH;
header = htonl(header);

temp = header;
for(i=0;i<4;i++){
    fprintf(f,"%c",(char)temp);
    temp = temp>>8;
}
fprintf(f,"0");
```

Uniquement en big endian

## Encoder une structure de donnée

```
uint32_t header = 1;
header = (header<<1);
header = header+1;
header = (header<<23);
header = header+LENGTH;
header = htonl(header);

temp = header;
for(i=0;i<4;i++){
    fprintf(f,"%c",(char)temp) ;
    temp = temp>>8;
}
fprintf(f,"0") ;
```

## Encoder une structure de donnée

```
uint32_t header = 1;  
header = (header<<1);  
header = header+1;  
header = (header<<23);  
header = header+LENGTH;  
header = htonl(header);  
  
write(fileno(f), &header, sizeof(header));
```

## Encoder une structure de donnée

```
uint32_t header = 1;  
header = (header<<1);  
header = header +1;  
header = (header<<23);  
header = header +LENGTH;  
header = htonl(header);  
  
write(fileno(f), &header,  
sizeof(header));
```

Vérifiez la longueur des champs!

LENGTH & 0xffff

F & 0x01

TYPE & 0x7fff



## Encoder une structure de donnée

```
uint32_t header = 1;
header = (header << 1);
header = header+1;
header = (header << 23);
header = header+LENGTH;
header = htonl(header);

write(fileno(f), &header,
sizeof(header));
```

En little endian :

```
1      == 0x01 0x00 0x00 0x00
1<<1  == 0x02 0x00 0x00 0x00
1<<1+1 ==
0x03 0x00 0x00 0x00
(1<<1+1)<<23 ==
0x00 0x00 0x80 0x01
htonl((1<<1+1)<<23) ==
0x01 0x80 0x00 0x00
```

Résultat voulu :

```
0x00 0x03 0x00 0x00
```

## Pointeur vers une structure temporaire

```
struct addrinfo *hints = malloc(sizeof(struct addrinfo));
struct addrinfo *res, *p;

memset(hints, 0, sizeof(struct addrinfo));
hints->ai_family = AF_INET6;
hints->ai_protocol = IPPROTO_UDP;
hints->ai_socktype = SOCK_DGRAM;
hints->ai_flags = AI_PASSIVE;

getaddrinfo(address, NULL, hints, &res);
...
free(hints);
```

## Pointeur vers une structure temporaire

```
struct addrinfo hints;  
struct addrinfo *res, *p;  
  
memset(&hints, 0, sizeof(struct addrinfo));  
hints.ai_family = AF_INET6;  
hints.ai_protocol = IPPROTO_UDP;  
hints.ai_socktype = SOCK_DGRAM;  
hints.ai_flags = AI_PASSIVE;  
  
getaddrinfo(address, NULL, &hints, &res);  
...
```

# Envoyer-recevoir des données

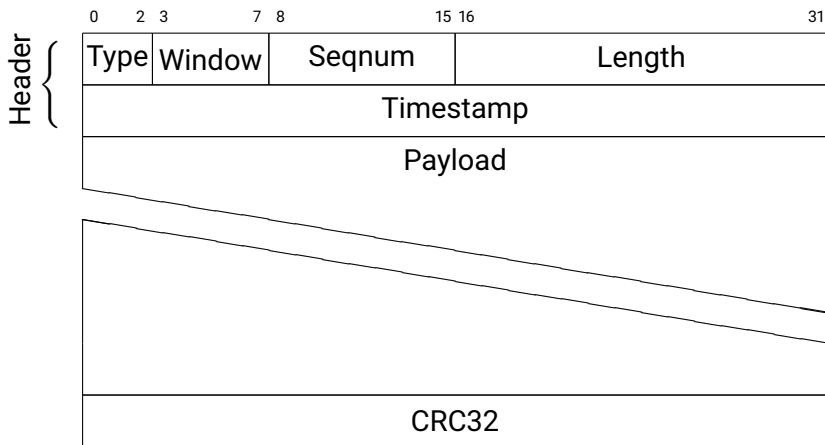
- Rappel : `fwrite` et `fread` font des opérations sur un buffer.
- `select()` *modifie* les `fd_set` donnés!
- `recvfrom` supporte plusieurs modes d'opération.
- Utilisez `fileno(stdin)` plutôt que `STDIN_FILENO` pour obtenir le file descriptor inclut dans un `FILE*` (e.g., pour résister à `freopen(..., stdin)`)

# Initialiser une structure

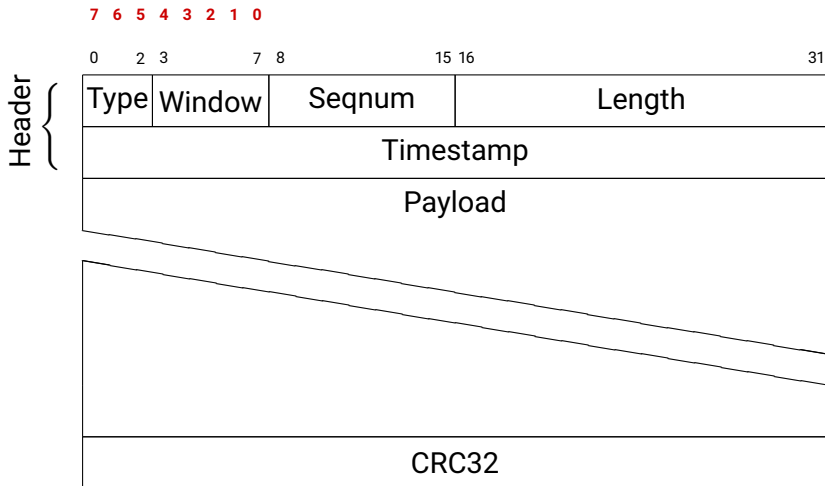
```
pkt_t* pkt_new()
{
    pkt_t *pkt =(pkt_t *)
        malloc(sizeof(pkt_t));
    if(pkt == NULL)
        return NULL;
    pkt->payload = NULL;
    return pkt;
}
```

```
pkt_t* pkt_new()
{
    return calloc(1,
        sizeof(pkt_t));
}
```

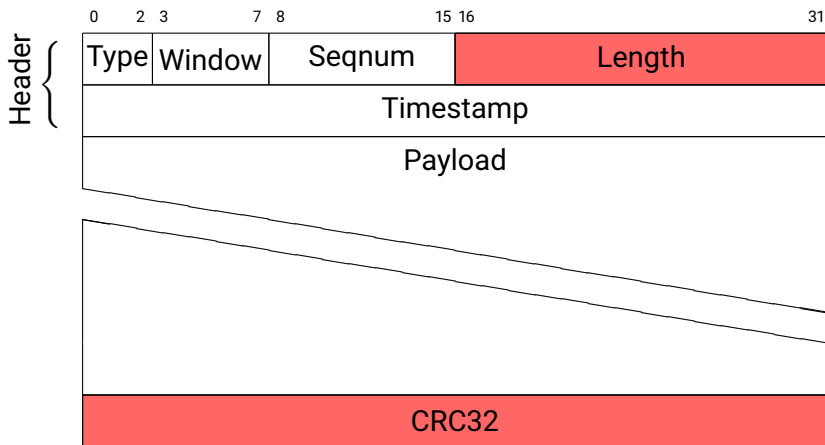
# Format des segments



# Numérotation des bits au sein d'un même octet



# Champs sensibles à l'endianness





# Spécifications du champs Timestamp

