**struct** rtp\_packet {

**int** version:2;

**int** padding:1;

**int** extension:1;

**int** ccount:4;

**int** marker:1;

**int** payloadtype:7;

**int** sequence:16;

**int** ssrc;

**int** timestamp;

**int** payload\_length;

**char** \*payload;

};

**void** \***create\_packet**(**void** \*p)

{

rtp\_packet \*ptr; [1]

rtp\_packet \*dummy;

ptr = (rtp\_packet \*) p; [3]

dummy = (rtp\_packet \*)malloc(1, **sizeof**(rtp\_packet \*)); [2]

**if** (dummy == NULL) {

**return** NULL;

}

\*dummy = \*ptr; [8]

free(ptr); [4]

ptr = dummy;

free(dummy); [5]

ptr->sequence ++; [6]

ptr->timestamp += 160; [7]

**return** ptr;

}

Remarks:

[1] Need to define those variables such as:

struct rtp\_packet\* ptr, \*dummy;

or

typedef struct \_rtp\_packet {

int version:2;

int padding:1;

int extension:1;

int ccount:4;

int marker:1;

int payloadtype:7;

int sequence:16;

int ssrc;

int timestamp;

int payload\_length;

} rtp\_packet;

[2] Incorrect use of malloc function.

**sizeof**(rtp\_packet \*) => this is the size of pointer and not the struct.

Need to write:

ptr = (struct rtp\_packet \*) p;

dummy = (struct rtp\_packet \*)malloc(sizeof(struct rtp\_packet));

or if we defined the struct at blue above:

ptr = (struct rtp\_packet \*) p;

dummy = (struct rtp\_packet \*)malloc(sizeof(struct rtp\_packet));

[3] didn’t check if p isn’t NULL pointer.

if (!p) {

return NULL;

}

[4] doesn’t sure if free the packet was the intention. Moreover we have

memory leak because payload field wasn’t released (if it wasn’t NULL).

[5] free the packet we have just allocated. Not the right behavior.

[6-7] we have just released this memory. We can’t access to it anymore.

Need to check if we didn’t exceed the size of sequence field (it’s 16

Bits size) before increasing it.

160 should be defined as MACRO.

#define DELAY 160

[8] copying the bits as is to dummy. Include copying the pointer of payload

from ptr. The right way is to allocate memory of payload field at dummy

variable if needed (maybe the pointer of payload at ptr is NULL). After

we allocate the needed space (with extra for NULL termination) copy the

string with strcpy or other string.h function.