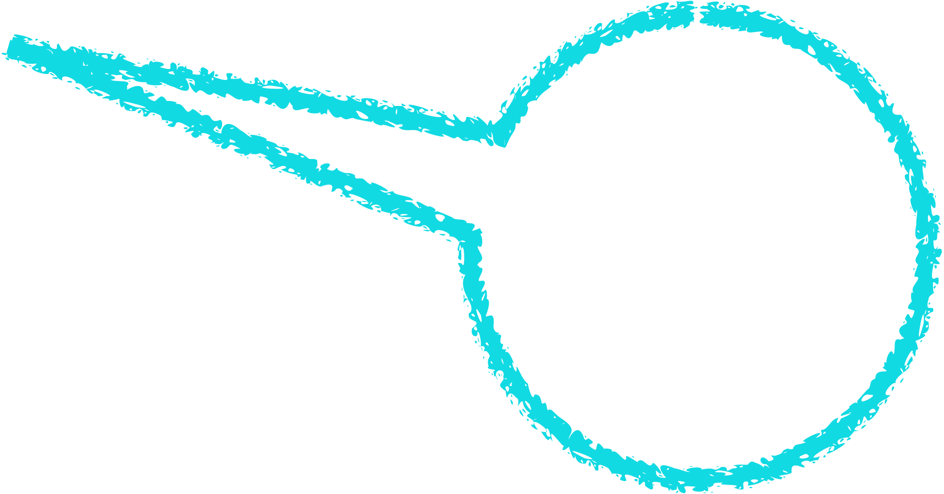
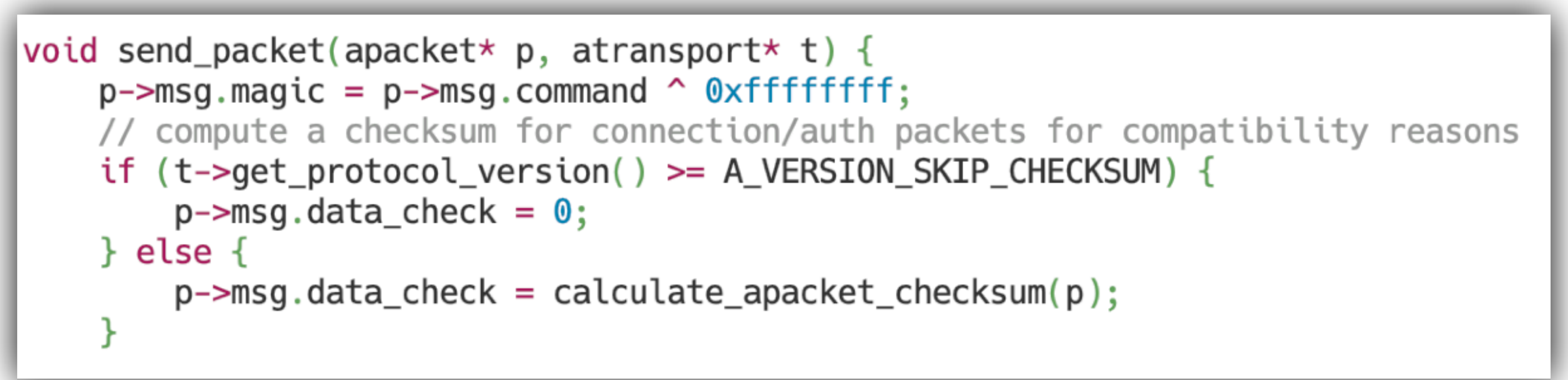
semantics of message



arguments

not used since Dec 2017



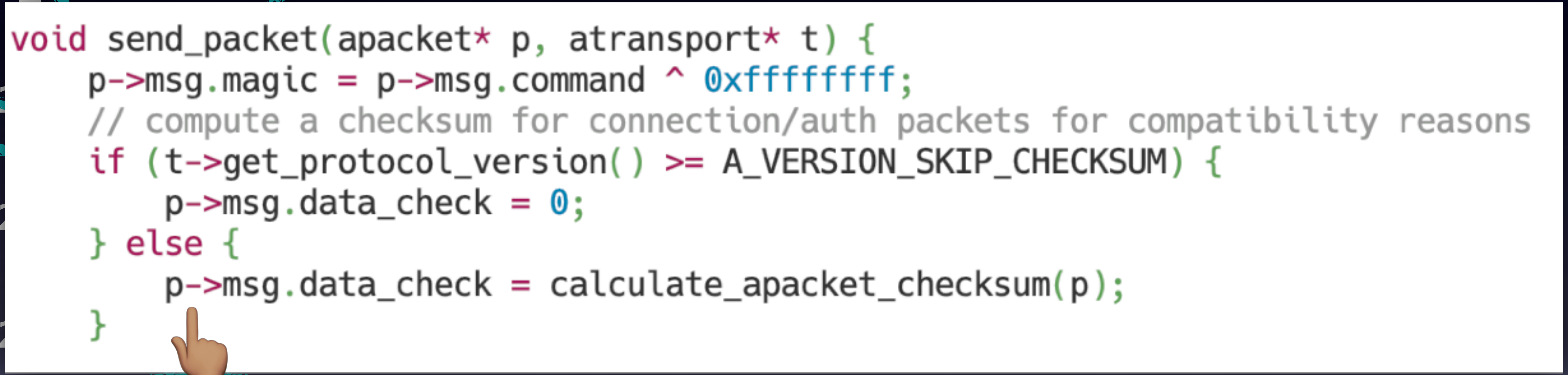
```
void send_packet(apacket* p, atransport* t) {
    p->msg.magic = p->msg.command ^ 0xffffffff;
    // compute a checksum for connection/auth packets for compatibility reasons
    if (t->get_protocol_version() >= A_VERSION_SKIP_CHECKSUM) {
        p->msg.data_check = 0;
    } else {
        p->msg.data_check = calculate_apacket_checksum(p);
    }
```

protocol message format

semantics of message

arguments

```
struct amessage {

    uint32_t command;     /* command identifier constant      */

    uint3

    uint3

    uint3

    uint32_t data_check;  /* checksum of data payload         */

    uint32_t magic;       /* command ^ 0xffffffff             */

};
```



```
void send_packet(apacket* p, atransport* t) {
    p->msg.magic = p->msg.command ^ 0xffffffff;
    // compute a checksum for connection/auth packets for compatibility reasons
    if (t->get_protocol_version() >= A_VERSION_SKIP_CHECKSUM) {
        p->msg.data_check = 0;
    } else {
        p->msg.data_check = calculate_apacket_checksum(p);
    }
```

not used since Dec 2017

message types

Code constant	Name	Use
A_CNXN	CONNECT	Setup a connection for a new client request
A_AUTH	AUTH	Authenticate a client to a device
A_OPEN	OPEN	Start a new stream
A_OKAY	READY	Indicate that far side is ready to accept more data
A_WRTE	WRITE	Send data for a specific stream
A_CLSE	CLOSE	Close a specific stream; sending side done or no more data available

32bitdesi (Ravi Subramanian)
in linkedin.com/in/ravisu @ravisu on Twitter