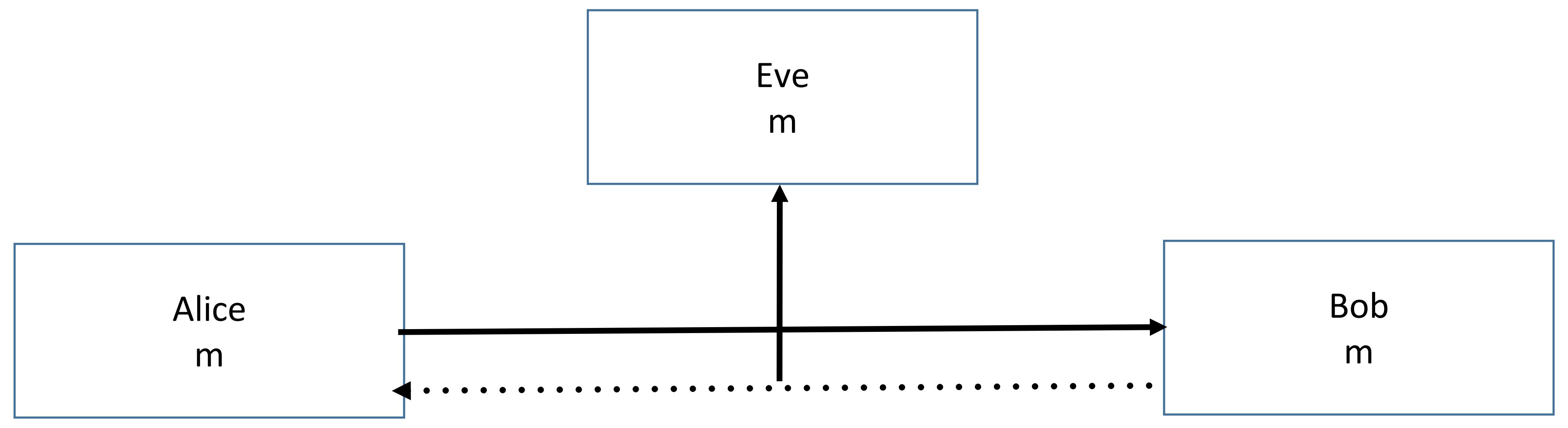
Class 7

The Secure Channel

General problem

Alice and Bob would like to create a secure connection



Roles:

Bob —> client

Alice —> server

Key (K):

Shared key, known to Bob and Alice known to them only

A new value is generated for the key per secure channel establishment

Session Key **K** is used for a single communication.

The security level 128 bits; thus a Key size of 256 bits is required

Distributing data among entities

Streams of data that can be distributed in a discrete fashion are only considered.

A data stream is then separated into discrete messages which will be assembled at the receiver side.

Transport system (layer) between Alice and Bob it is assumed to be not reliable. (From cryptographic point of view)

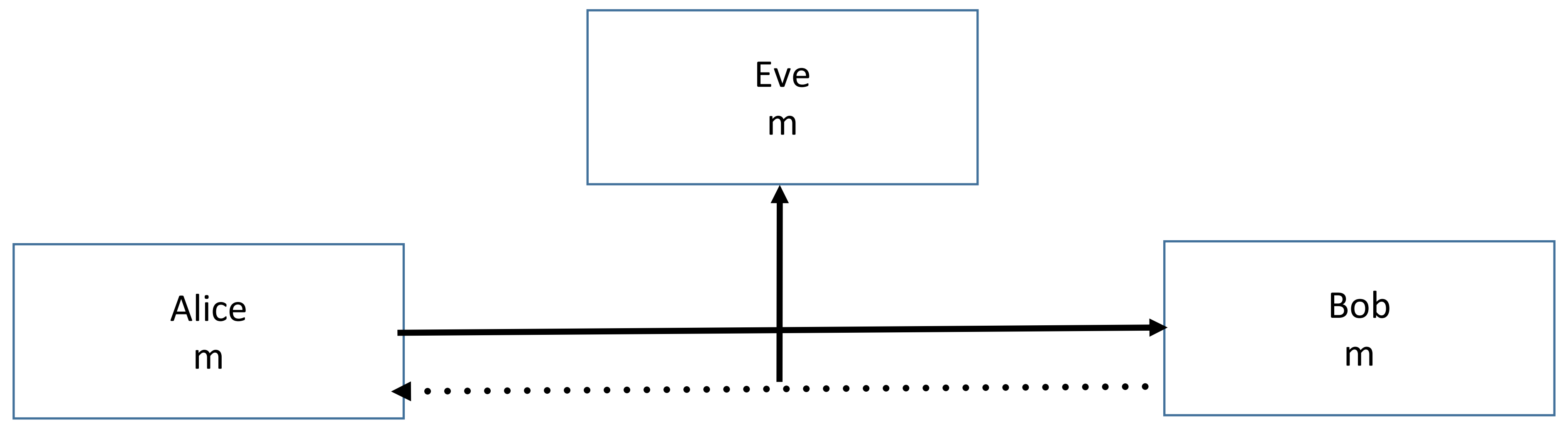
Distributing data among entities

From a cryptographic approach TCP/IP does not create a reliable communication channel. Cryptographers a reliable communication protocol does not exists.

Security Properties

Alice send to Bob

$M=\{m_1,m_2,\ldots,m_n\}$



Bob receives from Alice

$M'=\{m'_1,m'_2,\ldots,m'_n\}$

Security Properties: secrecy

Difficult to achieve, Eve can monitor secured data traffic (size and timing) among Alice and Bob. She could learn who is communication with whom, when and how much (traffic analysis)

Well know problem with secure channels like

SSL/TLS, IPSec and SSH.

Security Properties: notion of order

Alice will send a numbered sequence of messages to Bob. He will expect a subset of such sequence.

Messages could be dropped or intercepted by Eve.

Bob will not expect that messages lost will be send again by Alice.

This is not implemented in the secure channel, not incentive to resend lost messages.

Authentication and Encryption

Three possible approaches:

- **encrypt then authenticate: encryption and authentication can be done in parallel**

- authenticate then encrypt: **what would be more relevant if Eve is able to see the message or if she is able to modify it.?**

- encrypt the message then authenticate then concatenate results and then authenticate and then encrypt

A secure channel design

As a part of the design three components needs to be taken into account: message numbering, authentication and encryption.

Message Numbers 如果

Must increase monotonically and must be unique

Alice will assign a 1 to the first message, 2 to the second ..

A 32 bit number used for the message number for the secure channel. Limited by 2^{32} -1

Authentication

Choose a MAC for the authentication function. HMAC-SHA-256 with the full 256-bit. MAC input (m_i , x_i)

x_i describes contextual information i. e. protocol, files negotiated and protocol version among others. Alice and Bob will share the same string x_i

The Mac value ***a*** is calculated as:

$$a_i = MAC(i \,||\, l(x_i) \,||\, x_i \,||\, m_i)$$

Where ***i*** and ***l(x)*** are both 32-bit

Length is calculated as ***l(***x_i***)***

Encryption

AES in CTR mode is used, note the nonce is properly handled by the secure channel.

Size of the message limited 16 . 2^{32} bytes, in turns the block counter is limited by 32 bits.

Key stream is , k_0, $k_{1,}$ .... for a message with nonce *i*

$$k_0,\ldots,k_{2^{32}-1} := E(K, 0 || i || 0) || E(K, 1 || i || 0) || \ldots || E(K, 2^{32}-1 || i || 0)$$

Plaintext block 32-bit block number, 32 bit message number and 64 bits of zeros.

Only used the first $l(m_i)$+32 bytes

Concatenation of $\mathbf{m_i}$ and $\mathbf{a_i}$ **are XOR with** $\mathbf{k_{0,\ldots,}k_{l(m_i)+31}}$

Frame Format

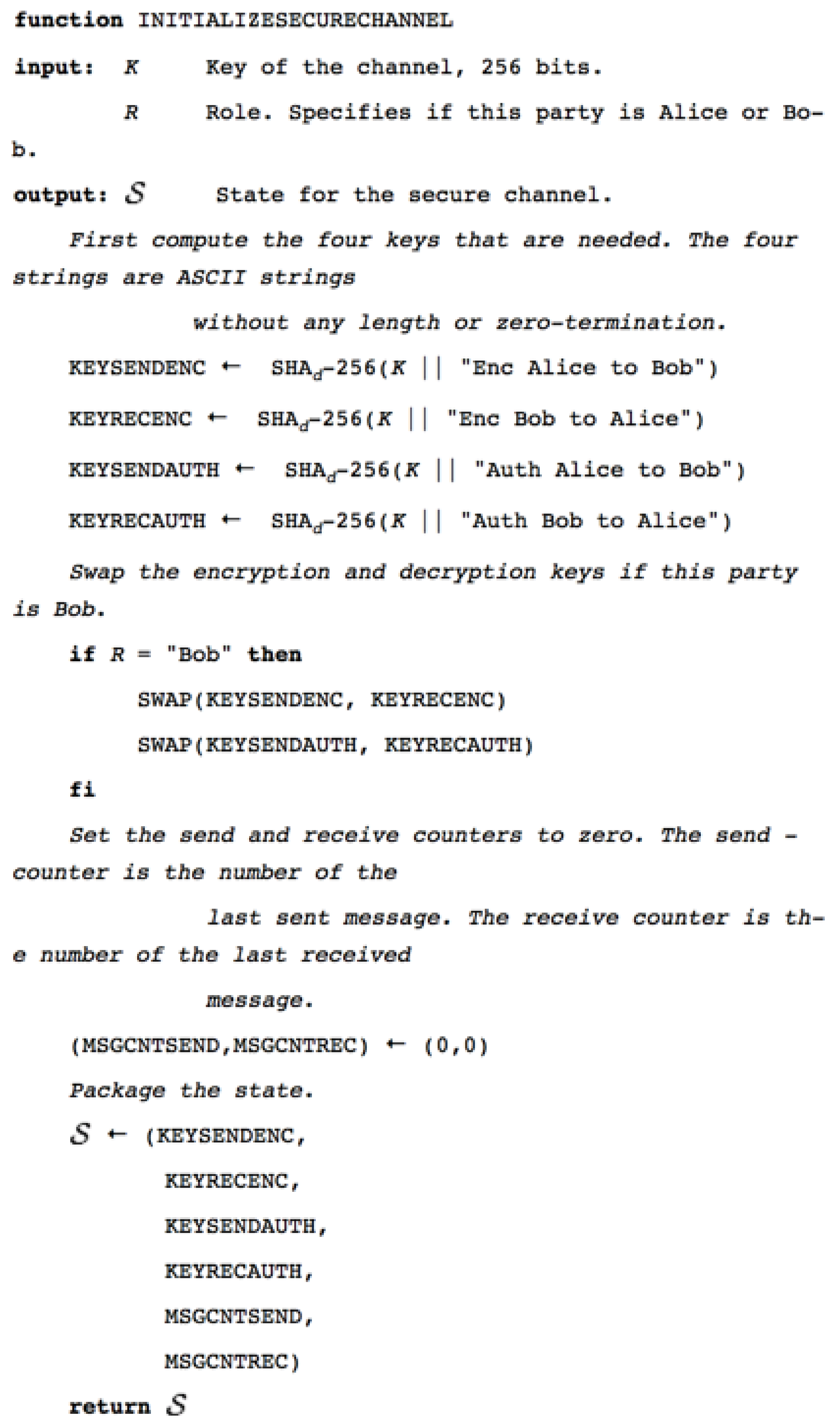
The message will be sent as i encoded as a 32 bit integer. First the least significant byte first, then the encrypted $\mathbf{m_i}$ and $\mathbf{a_i}$.

Initialization

Two functions:

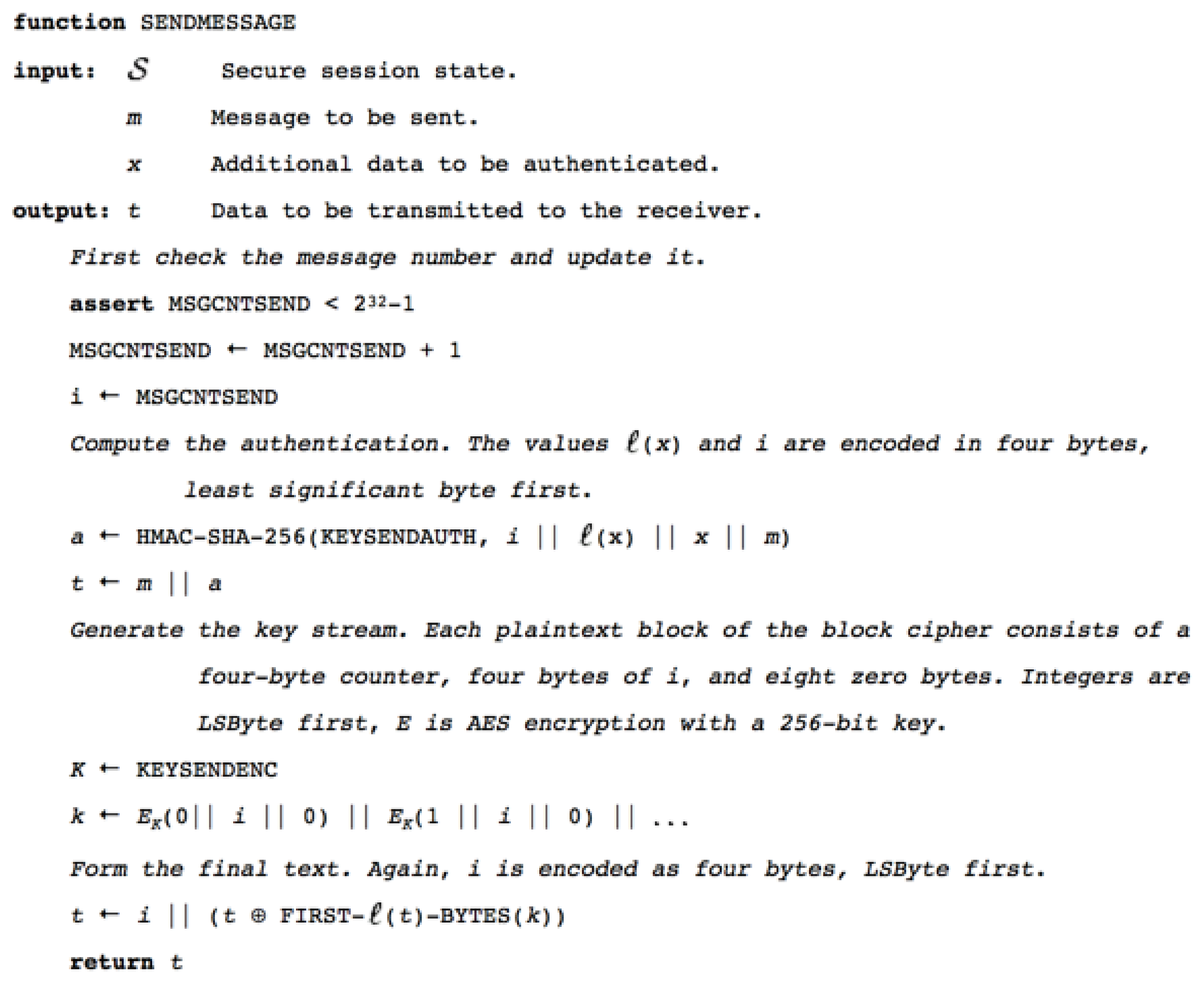
Setting the keys

Setting the message numbers



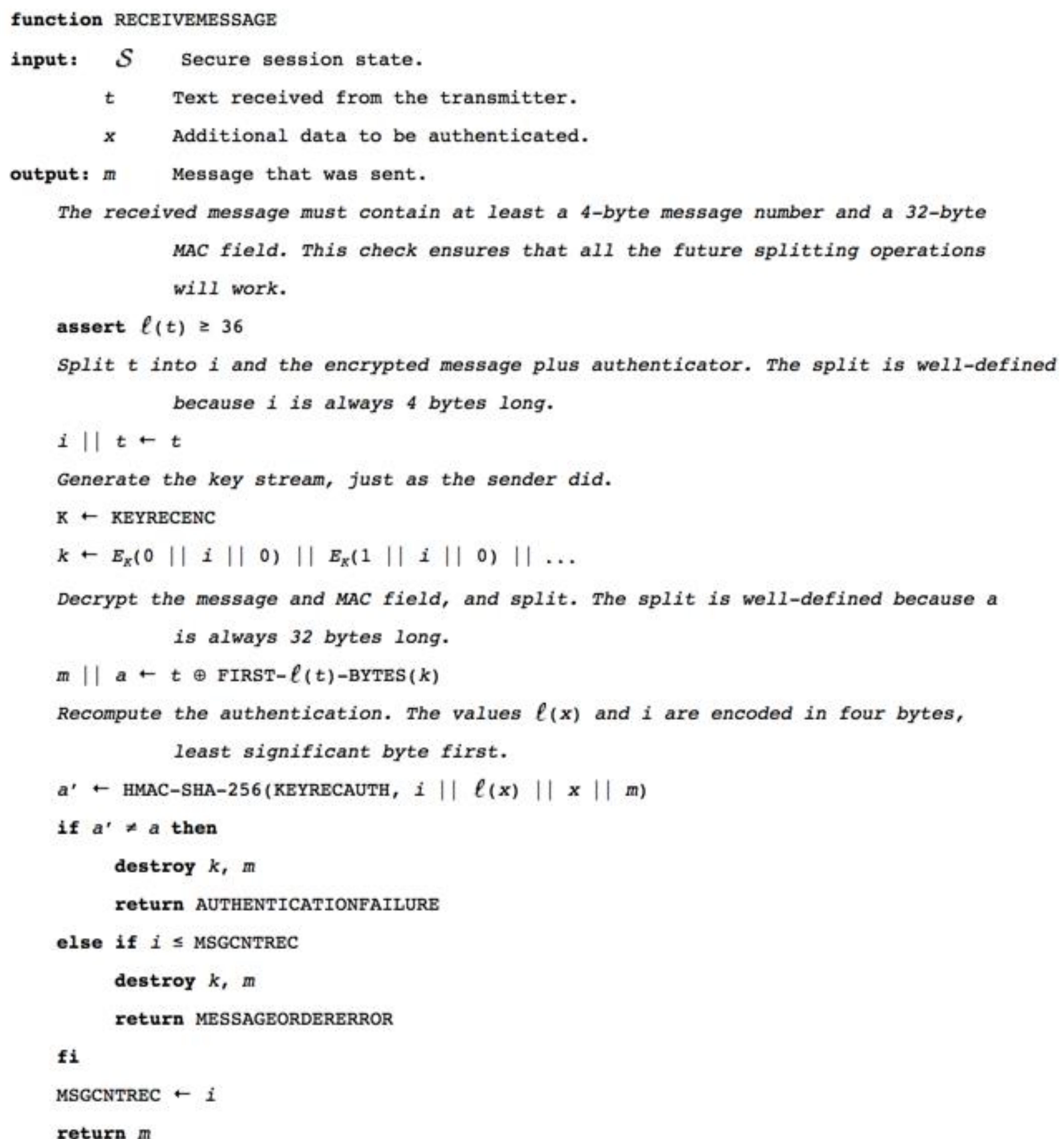
```
function INITIALIZESECURECHANNEL
input:  K     Key of the channel, 256 bits.
        R     Role. Specifies if this party is Alice or Bo-
b.
output: S     State for the secure channel.
    First compute the four keys that are needed. The four
strings are ASCII strings
                without any length or zero-termination.
    KEYSENDENC ← SHA$_d$-256(K || "Enc Alice to Bob")
    KEYRECENC ← SHA$_d$-256(K || "Enc Bob to Alice")
    KEYSENDAUTH ← SHA$_d$-256(K || "Auth Alice to Bob")
    KEYRECAUTH ← SHA$_d$-256(K || "Auth Bob to Alice")
    Swap the encryption and decryption keys if this party
is Bob.
    if R = "Bob" then
        SWAP(KEYSENDENC, KEYRECENC)
        SWAP(KEYSENDAUTH, KEYRECAUTH)
    fi
    Set the send and receive counters to zero. The send -
counter is the number of the
                last sent message. The receive counter is th-
e number of the last received
                message.
    (MSGCNTSEND,MSGCNTREC) ← (0,0)
    Package the state.
    S ← (KEYSENDENC,
          KEYRECENC,
          KEYSENDAUTH,
          KEYRECAUTH,
          MSGCNTSEND,
          MSGCNTREC)
    return S
```

Sending a message



```
function SENDMESSAGE
input:  $\mathcal{S}$    Secure session state.
        $m$    Message to be sent.
        $x$    Additional data to be authenticated.
output: $t$    Data to be transmitted to the receiver.
    First check the message number and update it.
    assert MSGCNTSEND < $2^{32}-1$
    MSGCNTSEND ← MSGCNTSEND + 1
    i ← MSGCNTSEND
    Compute the authentication. The values $\ell(x)$ and $i$ are encoded in four bytes,
            least significant byte first.
    $a$ ← HMAC-SHA-256(KEYSENDAUTH, $i \,||\, \ell(x) \,||\, x \,||\, m$)
    $t \leftarrow m \,||\, a$
    Generate the key stream. Each plaintext block of the block cipher consists of a
             four-byte counter, four bytes of $i$, and eight zero bytes. Integers are
             LSByte first, $E$ is AES encryption with a 256-bit key.
    $K$ ← KEYSENDENC
    $k \leftarrow E_K(0 \,||\, i \,||\, 0) \,||\, E_K(1 \,||\, i \,||\, 0) \,||\, \ldots$
    Form the final text. Again, $i$ is encoded as four bytes, LSByte first.
    $t \leftarrow i \,||\, (t \oplus$ FIRST-$\ell(t)$-BYTES$(k))$
    return $t$
```

Message reception



```
function RECEIVEMESSAGE
input:  $\mathcal{S}$   Secure session state.
        $t$   Text received from the transmitter.
        $x$   Additional data to be authenticated.
output: $m$   Message that was sent.
    The received message must contain at least a 4-byte message number and a 32-byte
            MAC field. This check ensures that all the future splitting operations
            will work.
    assert $\ell(t) \geq 36$
    Split t into i and the encrypted message plus authenticator. The split is well-defined
            because i is always 4 bytes long.
    $i \,||\, t \leftarrow t$
    Generate the key stream, just as the sender did.
    $K \leftarrow$ KEYRECENC
    $k \leftarrow E_K(0 \,||\, i \,||\, 0) \,||\, E_K(1 \,||\, i \,||\, 0) \,||\, \ldots$
    Decrypt the message and MAC field, and split. The split is well-defined because a
            is always 32 bytes long.
    $m \,||\, a \leftarrow t \oplus$ FIRST-$\ell(t)$-BYTES($k$)
    Recompute the authentication. The values $\ell(x)$ and i are encoded in four bytes,
            least significant byte first.
    $a' \leftarrow$ HMAC-SHA-256(KEYRECAUTH, $i \,||\, \ell(x) \,||\, x \,||\, m$)
    if $a' \neq a$ then
        destroy $k$, $m$
        return AUTHENTICATIONFAILURE
    else if $i \leq$ MSGCNTREC
        destroy $k$, $m$
        return MESSAGEORDERERROR
    fi
    MSGCNTREC $\leftarrow i$
    return $m$
```

Message order

Receiver checks message number it is expected to be increasing.

Order of arrival of the stream simplify the application implementation of the secure channel.

IPsec maintains a reply protection window instead of keeping track of the message number. IPsec uses a bit map if **d** is the message number of the last received message; d-31, d-30,…, d-1, d

what so important?

Alternatives to the secure channel

Dedicated use of block cipher modes:

CCM mode (Counter with CBC-MAC)

OCB mode (Offset Codebook Mode)

CWC Mode (Carter–Wegman + CTR mode)

GCM Mode (Galois Counter mode)

Among CCM, OCB, CWC and GCM, CCM and GCM are recommended. However they do not provide full secure channel

Adaptability of the secure channel will allow the use of this modes.