Project Assignment #1

(Frequency Evaluation)

SMP4PGMS

A Secure Multicast Protocol for a Peer-Group Messaging System

Context

- Initial materials[1] (provided) with na implementation of a PeerGroup Messaging System supported by IP Multicasting Communication
  - Connectionless multicast communication channel
  - Use of UDP (Datagram) Multicast Sockets
  - Insecure channel: possible attacks against communication
    - MiM attacks (Adversary model focusing on communication attacks)
      - Rationale: Analysis of possible attack types based on OSI X.800 Conceptual Framework

1. See:
http://vps726303.ovh.net/srsc2324/wa/

Goal

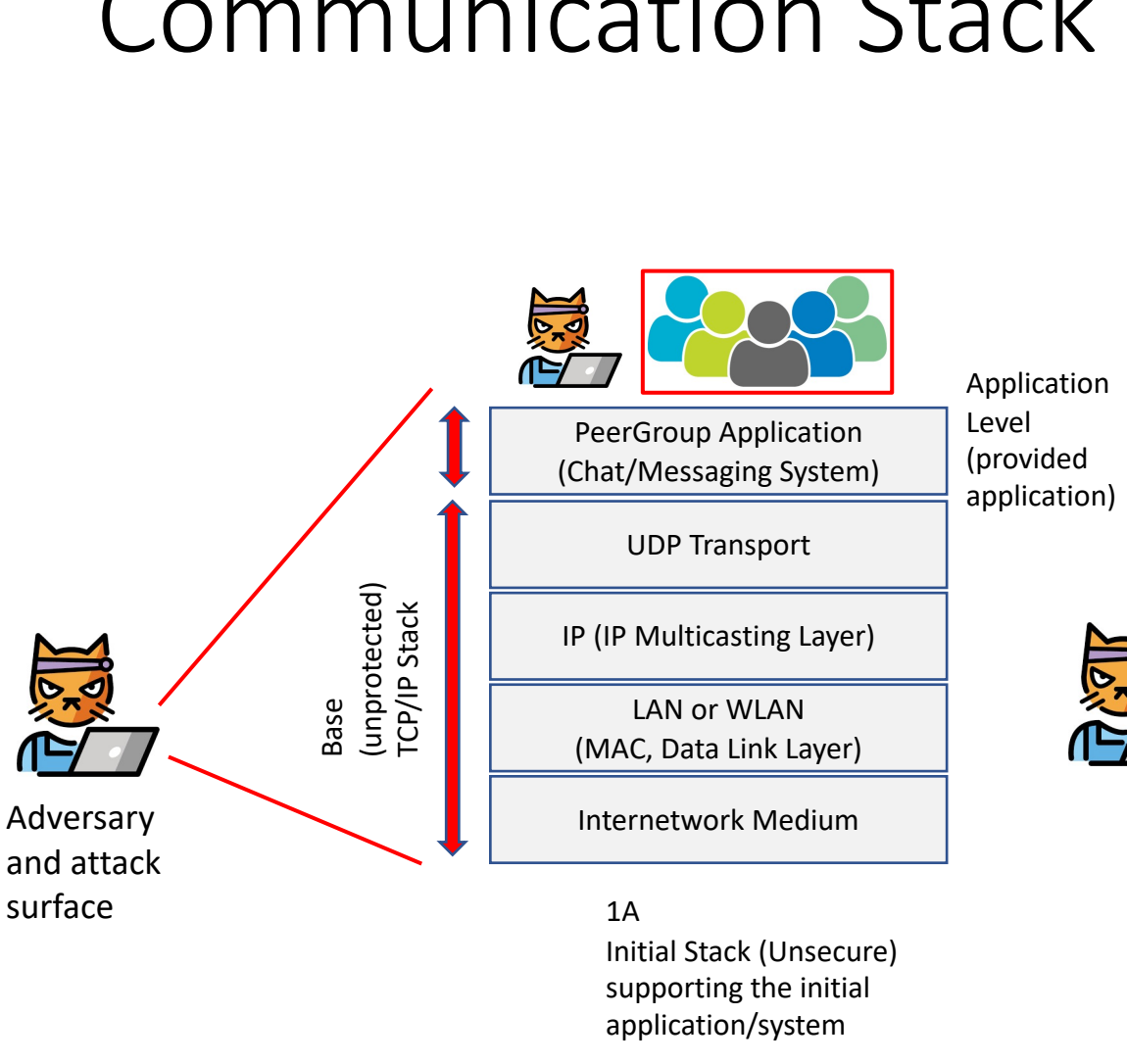
- Design, implementation, validation and experimental evaluation of a **Secure Multicast Protocol for a Peer-Group Messaging System (SMP4PGMS)**
  - **To protect the connectionless multicast communication channel supported by IP Multicast**

Idea: implementation of security properties as countermeasures against attack types (in the considered adversary model assumptions), protecting the communication channel (and supported interactions) between the group peers

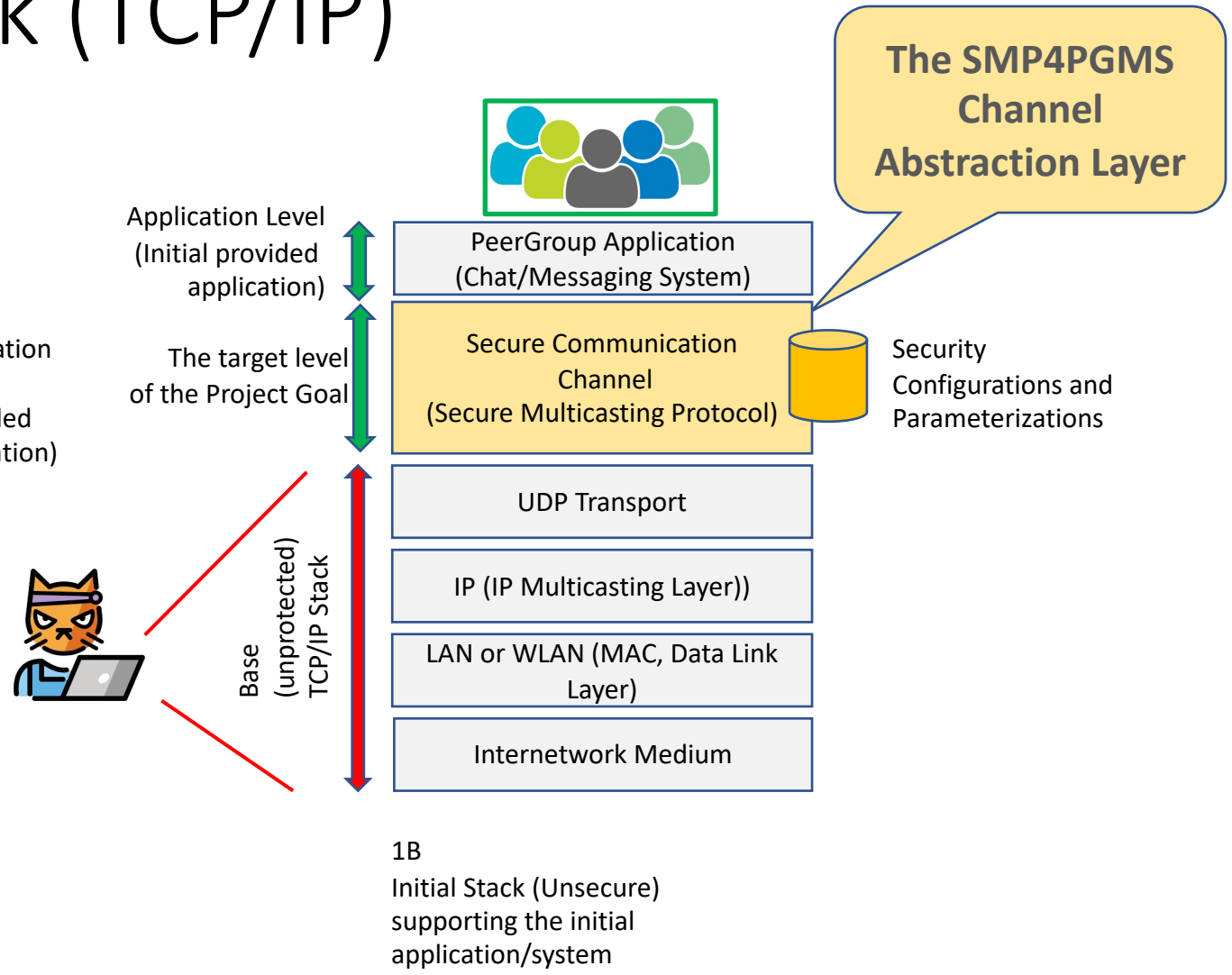
- Message confidentiality (connectionless confidentiality)
- Message integrity (connectionless integrity detection w/o recovery)
- Message authenticity
- Peer-Authenticity

How: implementing the SMP4PGMS with the required cryptographic constructions for the expected security guarantees

Communication Stack (TCP/IP)



1A
Initial Stack (Unsecure)
supporting the initial
application/system



1B
Initial Stack (Unsecure)
supporting the initial
application/system

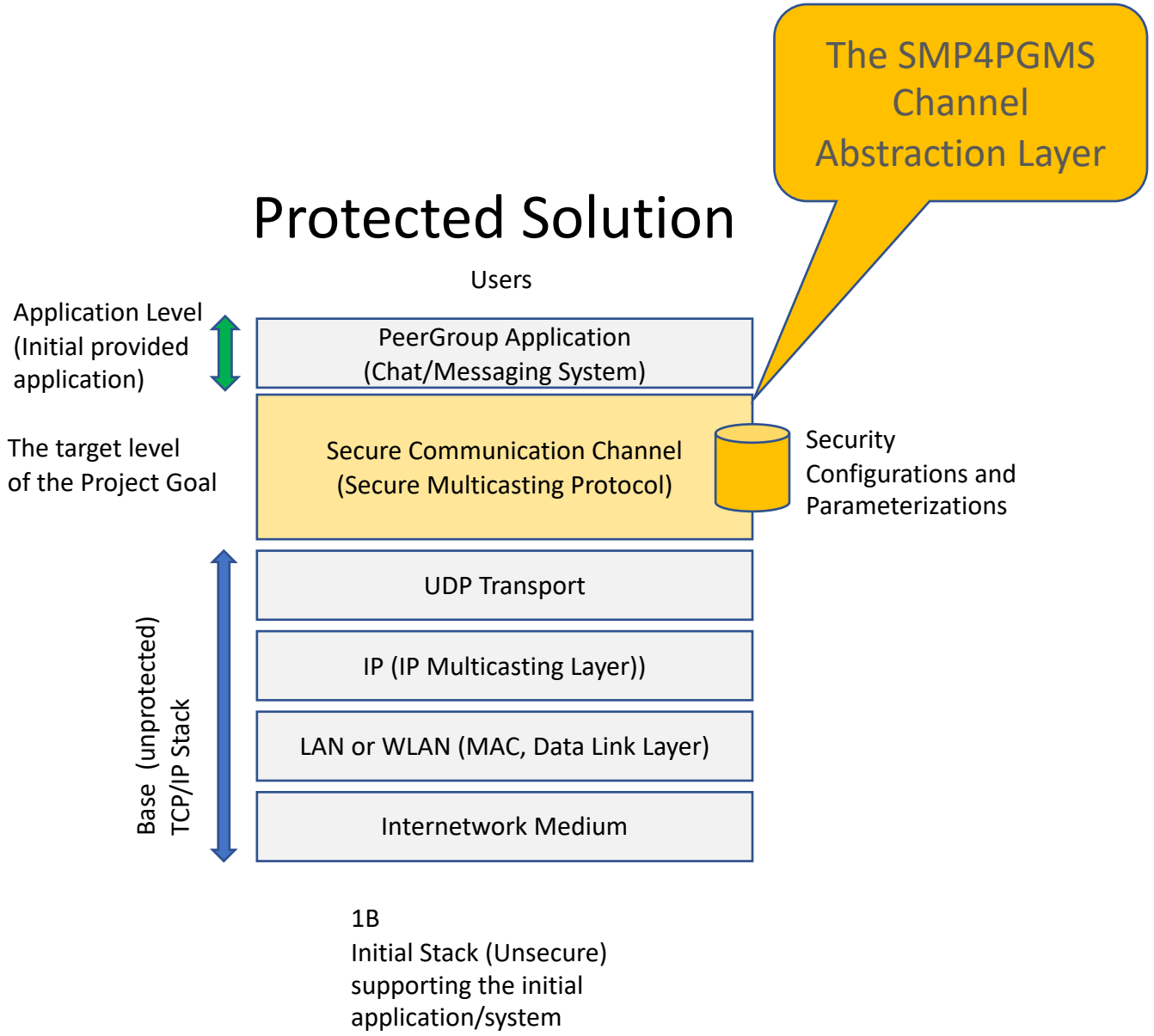
Unprotected Solution

Protected Solution

The required solution

Must support (transparently)
The Peer Group-Messaging
System as initially provided:
(*MChatClient, MulticastChatListener*)

Minimal and focused changes
In the initial solution
(*MulticastChat => SecureMulticastChat*)
Challenge:
diff (MulticastChat, SecureMulticastChat)
= Minimum LoC max as ref.
The minimum… The better



Implementation: in 4 phases as suggested (as incremental AGP sprints and milestones)

Phase 1: Static and Rigid Cryptographic Parameterizations

- Msg Confidentiality, Msg Authenticity and Msg Integrity

Phase 2: Static but Flexible Cryptographic Parameterizations

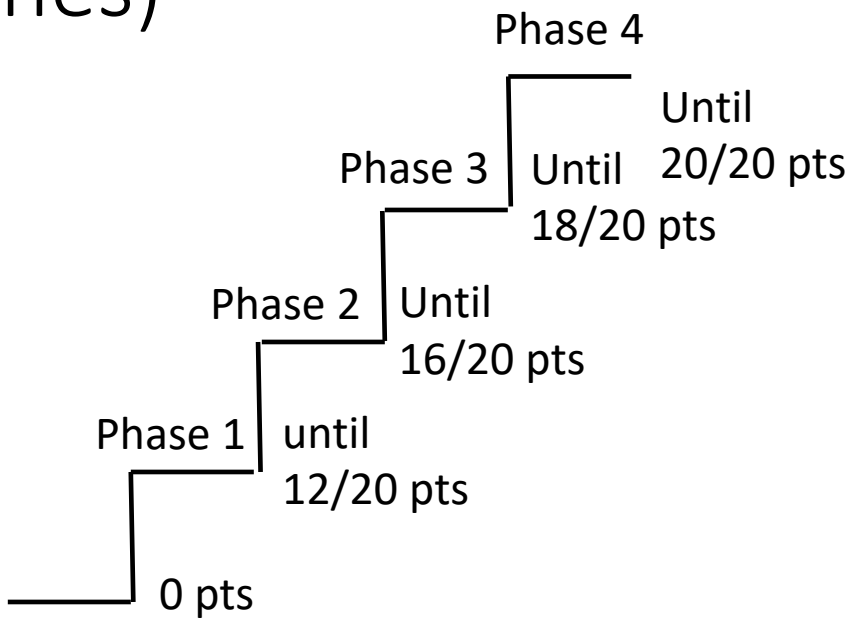
- Msg Confidentiality, Msg Authenticity and Msg Integrity

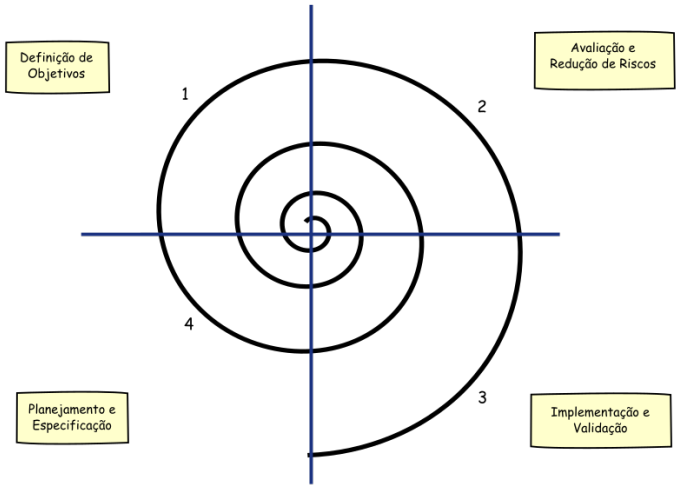
Phase 3: Extend Phase 2 with Peer-Authentication guarantees

- Msg Confidentiality, Msg Authenticity, Msg Integrity and Peer-Authenticity

Phase 4: Extend Phase 3 avoiding static configurations

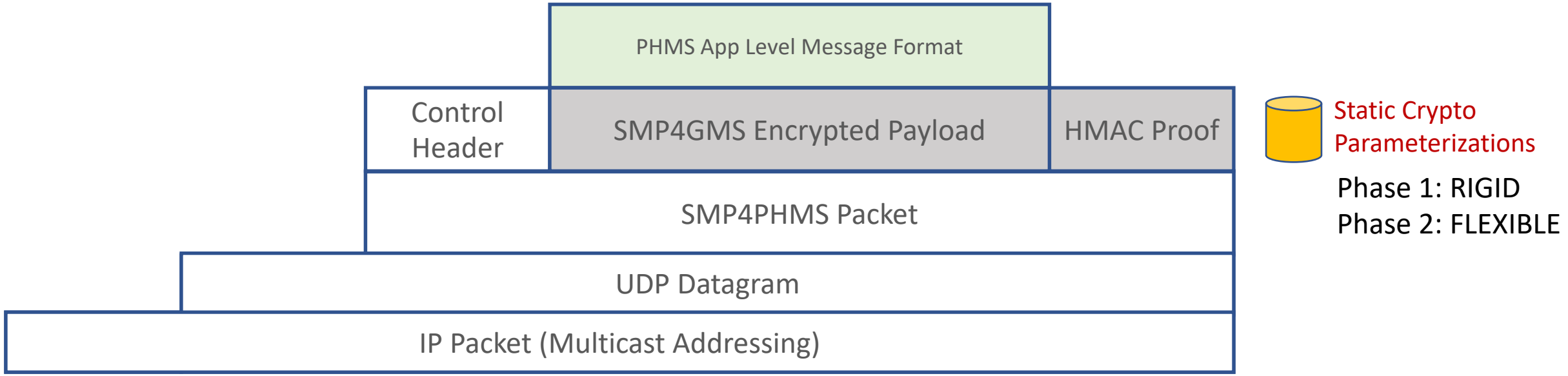
- All cryptographic parameterizations and related security association parameters will be setup dynamically by the protocol
- Minimal (or none at all) static crypto configurations





Implementation in **Phase 1 and Phase 2:** **SMP4PGMS** Message Format and Encapsulation

`Plaintext Control Header:  VERSION || CHAT_MAGIC_NUMBER || Hash (sender name)`
`SMP4GMS Encrypted Payload:` E_{Ks} `( SENDER_NAME || MSG_TYPE_TAG || NONCE || MSG_DATA )`
`- - -` Symmetric Enc. w/ or w/o Parameterized Mode w/ or w/o Padding `- - -`
HMAC Proof = $HMAC_{Km}$ (Plaintext Control Header || SMPGMS Encrypted Message)



SEND TO MAC / DATA-LINK LEVEL FRAME AND THEN TO THE PHYSICAL NETWORK

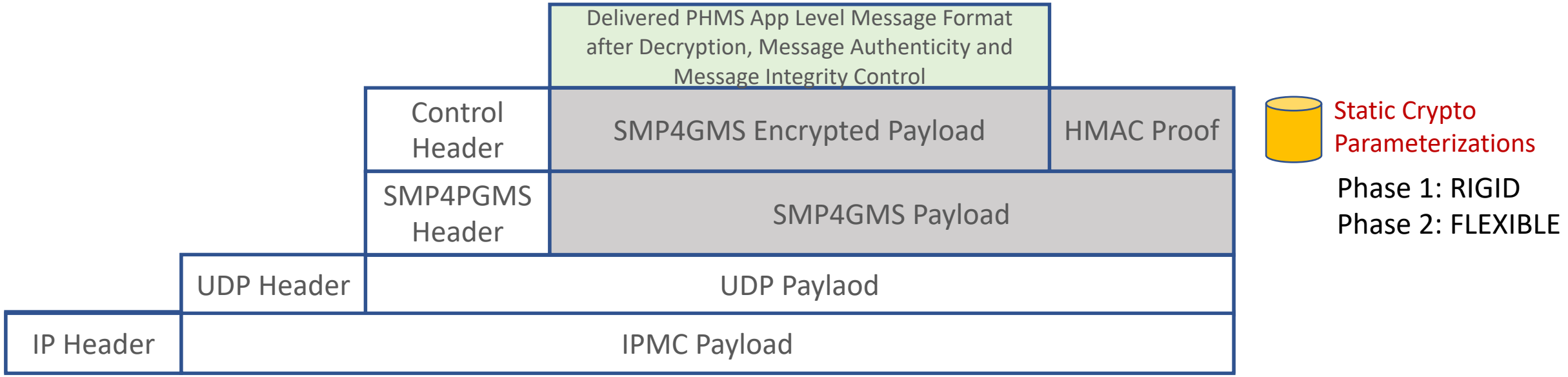
Implementation in **Phase 1 and Phase 2:** **SMP4PGMS** Message Format and Desencapsulation

`Plaintext Control Header:` `VERSION || CHAT_MAGIC_NUMBER || Hash (sender name)`

`SMP4GMS Encrypted Payload:` D_{Ks} `( SENDER_NAME || MSG_TYPE_TAG || NONCE || MSG_DATA )`

`- - -` Symmetric Enc. w/ or w/o Parameterized Mode w/ or w/o Padding `- - -`

HMAC Proof = $HMAC_{Km}$ (Plaintext Control Header || SMPGMS Encrypted Message)



RECEIVED FROM MAC / DATA-LINK LEVEL FRAME OBTAIND FROM THE PHYSICAL NETWORK

Implementation in **Phase 3**: SMP4PGMS Message Format and Encapsulation.

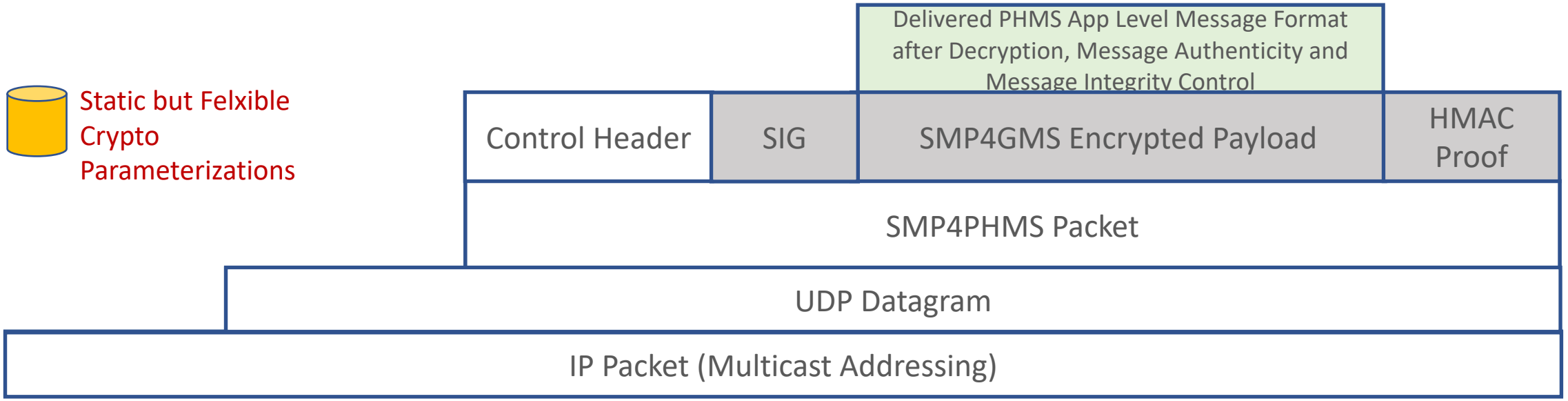
`Plaintext Control Header:  VERSION || CHAT_MAGIC_NUMBER || Hash (sender name)`

`SMP4GMS Encrypted Payload:` E_{Ks} `( SENDER_NAME || MSG_TYPE_TAG || NONCE || MSG_DATA )`

- - - Symmetric Enc. w/ or w/o Parameterized Mode w/ or w/o Padding - - -

SIG: $Signature_{PrivateKeySender}$ (Plaintext Control header || SENDER_NAME || MSG_TYPE_TAG || NONCE || MSG_DATA)

HMAC Proof = $HMAC_{Km}$ (Control Header || SIG || SMPGMS Encrypted Message)



SEND TO MAC / DATA-LINK LEVEL FRAME AND THEN TO THE PHYSICAL NETWORK

Implementation in **Phase 3**: SMP4PGMS Message Format and Desencapsulation

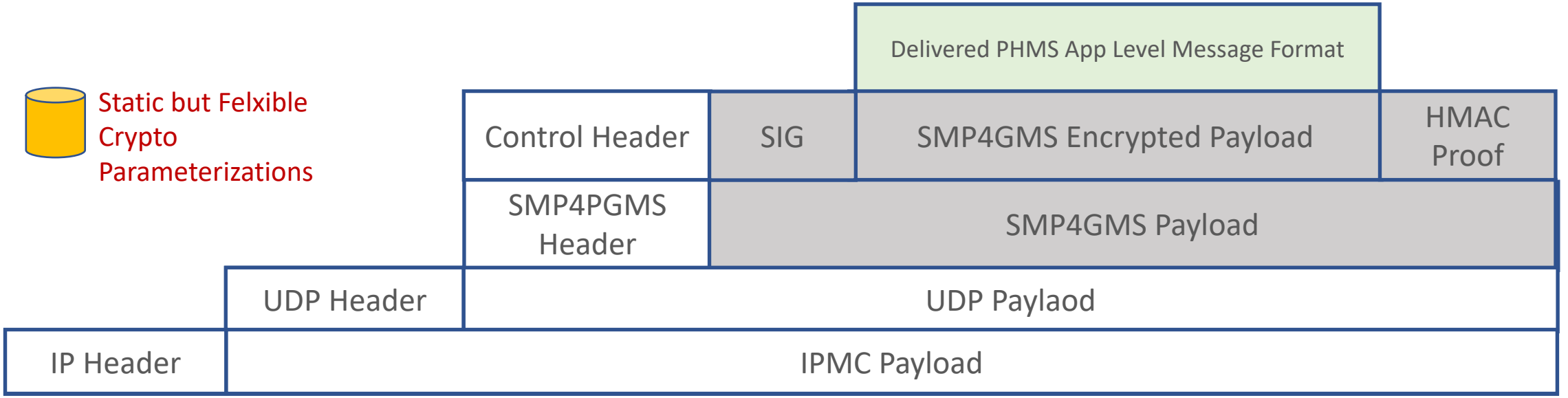
`Plaintext Control Header:` `VERSION || CHAT_MAGIC_NUMBER || Hash (sender name)`

`SMP4GMS Encrypted Payload:` D_{Ks} `( SENDER_NAME || MSG_TYPE_TAG || NONCE || MSG_DATA )`

- - - Symmetric Enc. w/ or w/o Parameterized Mode w/ or w/o Padding - - -

SIG: $ValidateSig_{PublicKeySender}$ (Plaintext Control header || SENDER_NAME || MSG_TYPE_TAG || NONCE || MSG_DATA)

HMAC Proof = $HMAC_{Km}$ (SMPGMS Encrypted Message || SIG || Plaintext Control Header)



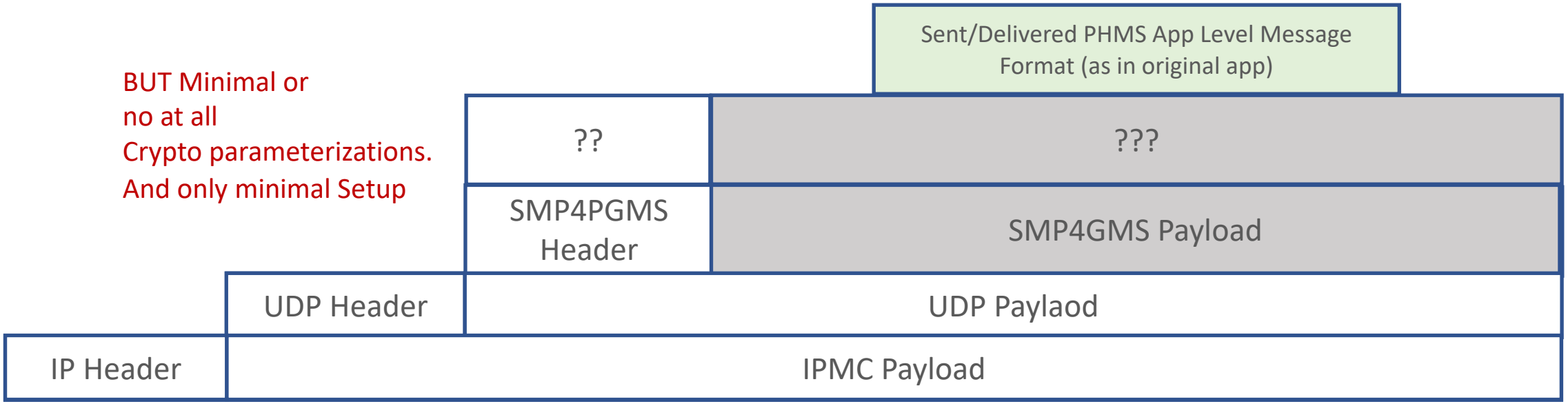
RECEIVED FROM MAC / DATA-LINK LEVEL FRAME AND FROM THE PHYSICAL NETWORK

Implementation in **Phase 4:** SMP4PGMS Message Format Encap/Desencap.

Challenges: Same as Phase 3: Flxible Cryptpgraphic Parameterizations ... But

... Need to establish dynamically Keys and all required crypto parameterizations

BUT Minimal or
no at all
Crypto parameterizations.
And only minimal Setup



SENT/RECEIVED TO/FROM MAC / DATA-LINK LEVEL FRAME AND FROM THE PHYSICAL NETWORK

More... (See the requirements)

- See the remaining requirements
  - Configurations (config files) for cryptographic parameterizations and setup for Phase 1 + tests/verifications
  - Configurations (config files) for cryptographic parameterizations and setup for Phase 2 + tests/verifications
  - Configurations (config files) for cryptographic parameterizations and setup for Phase 3 + tests/verifications

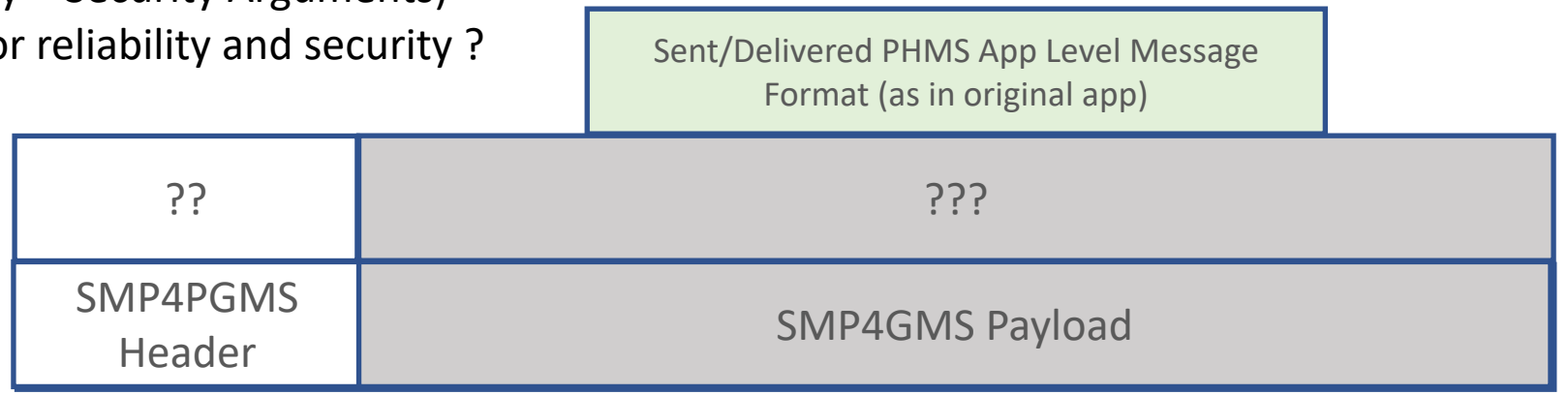
  - How to have minimal (or no at all) cryptographic parameterizations or minimal setup in Phase 4 + tests/verifications

Is there a Phase 5 ? ;-)

Challenges: It could be leveraged by Phase 4 … ... But w/ some other additional challenges, ex:

- How to address Traffic Flow Confidentiality and Traffic Flow Integrity in a Session-Oriented Model
- How to address PFS and PBS ?
- How to address dynamic refreshment of Keys : Encryption/Decryption Keys and HMAC Keys ?
- Can we address also dynamic refreshment of cryptographic parameterizations (ciphersuites and related security association parameters) ?
- Can we support the above in a “one-time basis per message” … or in a “one time basis per session” ?
- Uhm .. It is about a group-oriented model… Can we establish group-keys ?
- Refreshed group keys with PFS and PBS guarantees when each user joins/leave ?
- What more ??? (Reliability + Security Arguments)
  - What do we need for reliability and security ?

With minimal or no at all Parameterizations. Minimal Setup



SENT/RECEIVED TO/FROM UDP/IP Multicasting Channel