Temporal Key Integrity Protocol (TKIP)

Presented By:

Laxmi Nissanka Rao

Kim Sang Soo

Agenda

- Disadvantages of WEP
- Design Constraints
- Components of TKIP
- Putting the pieces together
- Questions

Disadvantages of WEP

- WEP provides no forgery protection
- No protection against Message Replays
- WEP misuses the RC4 encryption algorithm in a way that exposes the protocol to weak key attacks
- By reusing initialization vectors, WEP enables an attacker to decrypt the encrypted data without ever learning the encryption key

Design Constraints

- WEP-patches, on the already deployed hardware, have to depend entirely on software upgrades.
- The paucity of the CPU cycles.
- The hardwiring of the encryption algorithm.

TKIP

- Temporal Key Integrity Protocol (TKIP) is the TaskGroupi's solution for the security loop holes present in the already deployed 802.11 hardware
- It is a set of algorithms that wrap WEP to give the best possible solution given all the above mentioned design constraints.

Components of TKIP

- A cryptographic message integrity code, or MIC, called Michael: to defeat forgeries;
- A new IV sequencing discipline: to remove replay attacks from the attacker's arsenal;
- A per-packet key mixing function: to de-correlate the public IVs from weak keys
- A re-keying mechanism: to provide fresh encryption and integrity keys, undoing the threat of attacks stemming from key reuse.

Defeating Forgeries: Michael

- Every MIC has three components: a secret authentication key K (shared only between the sender and receiver), a tagging function, and a verification predicate.
- Designed by Niels Ferguson.

Michael (contd.)

- 64-bit Michael key: represented as two 32-bit words (K0,K1).
- The tagging function first pads a message with the hex value 0x5a and enough zero pad to bring the total message length to a multiple of 32-bits, then partitions the result into a sequence of 32-bit words M1,M2, … Mn.

$(L,R) \leftarrow (K0,K1)$

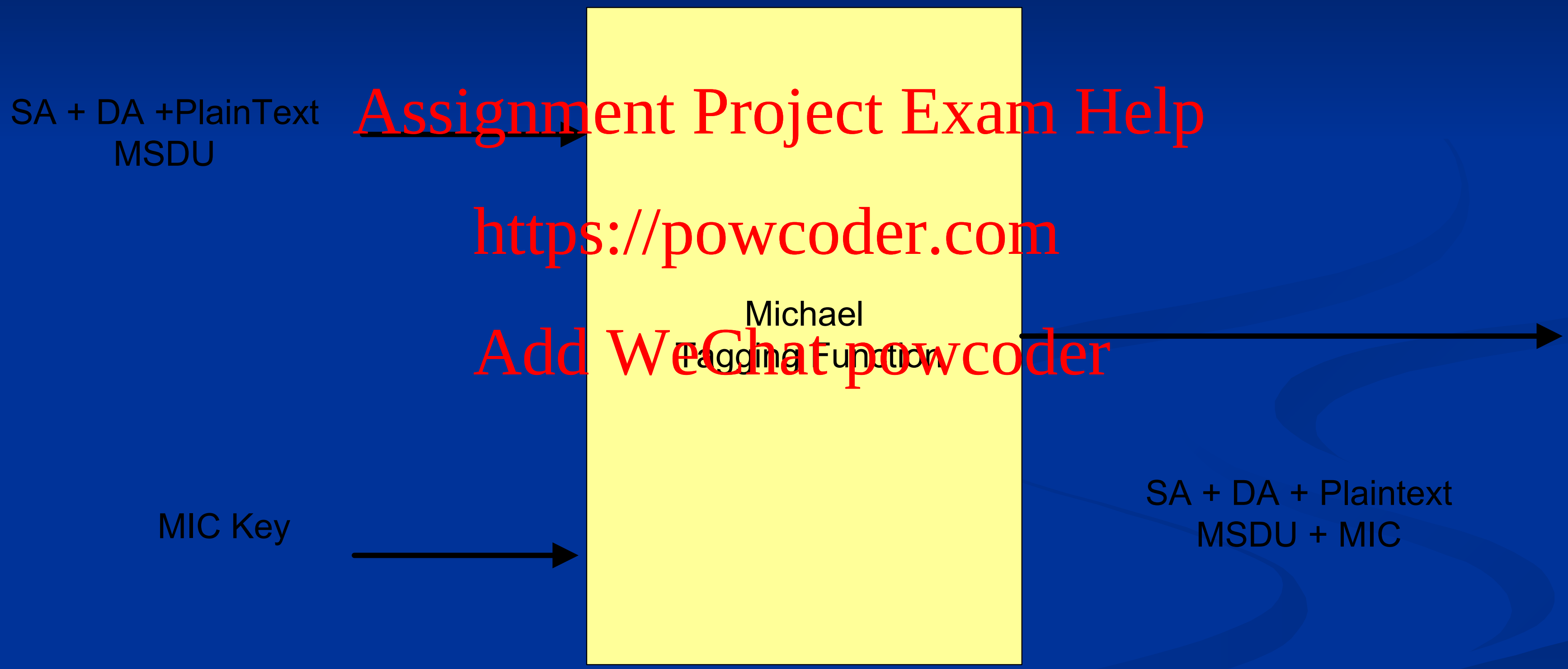
do i from 1 to n

  - $L \leftarrow L \wedge Mi$
  - $(L,R) \leftarrow b\ (L,R)$

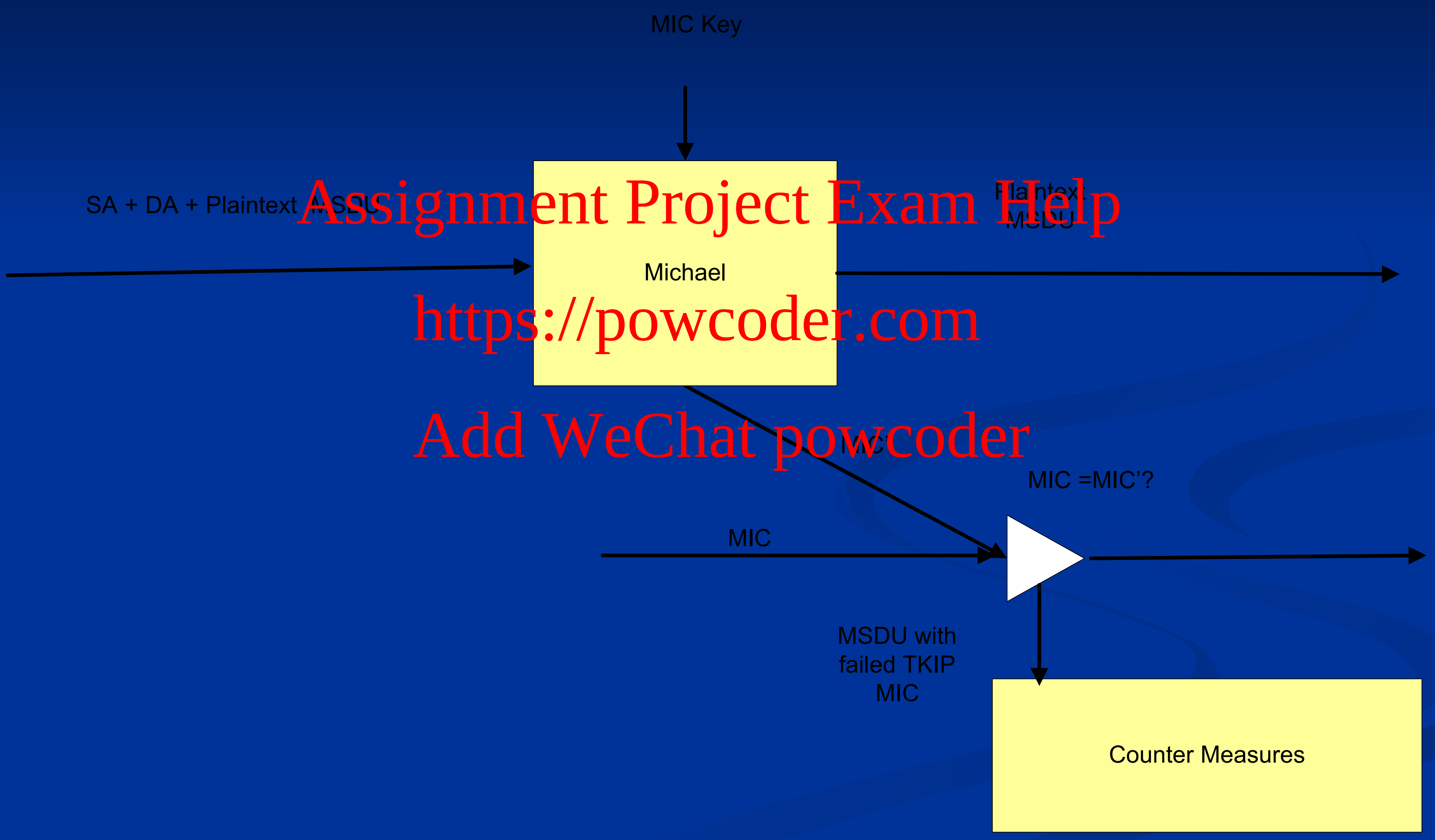
return (L,R) as the tag

- Where b is a function built up from rotates, little-Endean additions, and bit swaps.

Michael: Tagging Function



Michael: Verification Predicate



Michael (contd.)

- The design goal of the counter-measures is to throttle the utility of forgery attempts.
- If a TKIP implementation detects two failed forgeries in a second, the design assumes it is under active attack. The station deletes its keys, disassociates, waits for a minute, and then re-associates.

Defeating replays: IV sequence enforcement

- TKIP reuses the WEP IV field as a packet sequence number.
- Both transmitter and receiver initialize the packet sequence space to zero whenever new TKIP keys are set, and the transmitter increments the sequence number with each packet it sends.

IV sequence enforcement (contd.)

- TKIP defines a packet as out-of-sequence if its IV is the same or smaller than a previous correctly received MPDU associated with the same encryption key.
- If an MPDU arrives out of order, then it is considered to be a replay, and the receiver discards it and increments a replay counter.

Per-Packet Key Mixing

- **WEP** constructs a per-packet key by simply concatenating a base-key and the IV

- **TKIP** constructs a per-packet key by going through 2 key mixing phases

  - The mixing phases make difficult for an attacker to correlate IVs and per-packet key

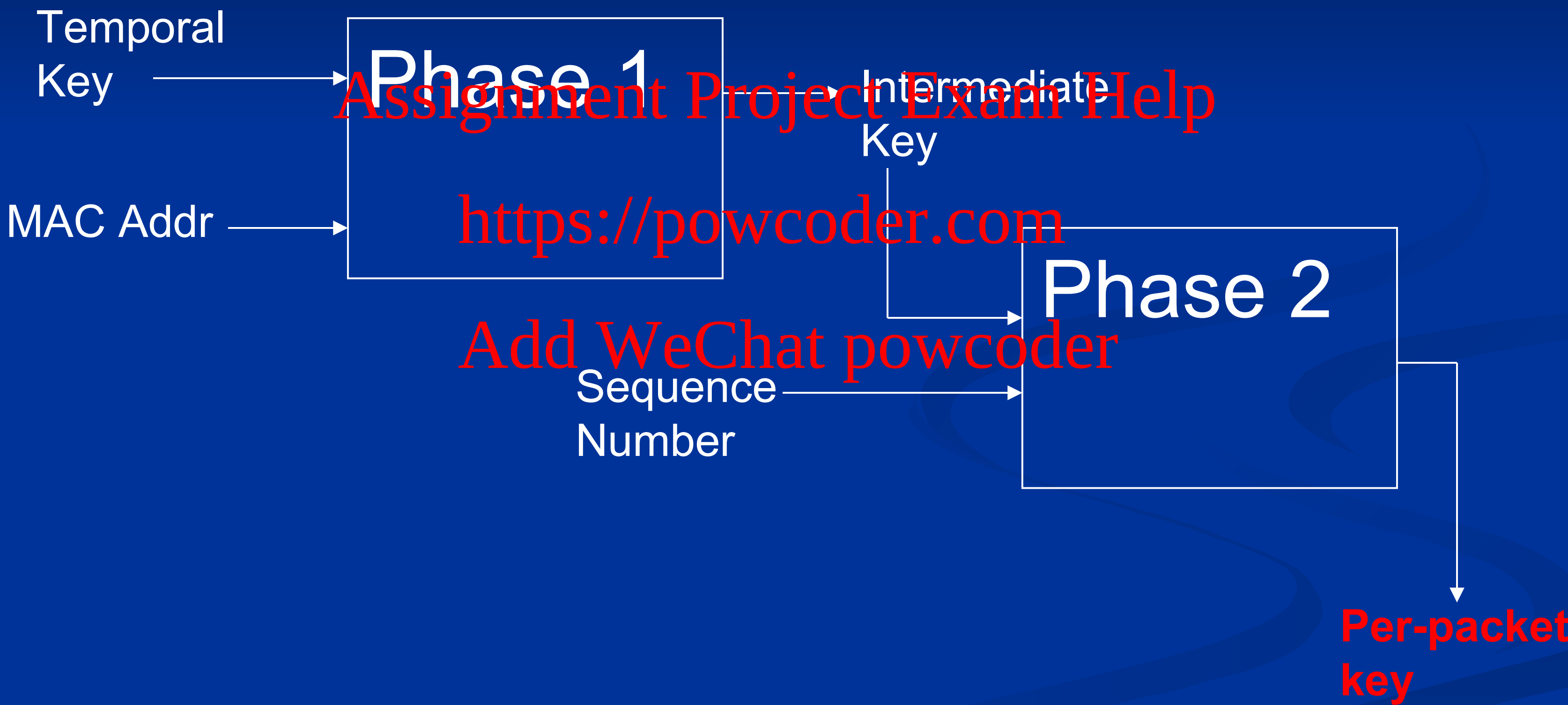
Per-Packet Key Mixing: 1st Phase

- XORs the **MAC address** of the station and the **temporal key** to produce an *intermediate key*
- Mixing MAC and the temporary key in this way causes different stations and APs to generate different *intermediate keys*, even if they have the same temporal key
- For performance optimization, *intermediate key* is computed only when the temporal key is changed (and most of the time its value is saved on memory)

Per-Packet Key Mixing: 2nd Phase

- Takes the packet sequence number and encrypts it using the *intermediate key* from the first phase, producing finally a 128-bit per-packet key
- In actuality, the first 3 bytes (24 bits) of Phase 2 output corresponds exactly to the WEP IV, and the last 13 bytes to the WEP base key.
- Now we can use the existing WEP hardware to do the encryption using the per-packet key

Per-Packet Key Mixing: Diagram



ReKey Mechanism

- Refers to a process of delivering fresh encryption and integrity keys (MIC Keys) to the stations and APs

- Accomplished by employing IEEE 802.1X

  - Defines an authentication server that distributes keys
- TKIP uses three distinct keys

  1. Temporal keys
  2. key encryption keys
  3. master keys

Temporal Keys

- Two Temporal Key types:
  - 128-bit encryption key
  - 64-bit Michael key
- Used by stations and APs for normal TKIP communication

Key Encryption Keys

- As the name suggests, a temporal key is “temporal” and needs to be updated frequently
- **Key Encryption Keys** encrypt the information regarding the key distribution. They protect the Temporal Keys.
- Requires two distinct key encryption keys
  1. To encrypt the distributed Keying material
  2. To protect the re-key messages from forgery

Master Key

- Used to secure the distribution of the key encryption keys
- Also related to TKIP's support of user authentication:
- A station gets a master key after it is "authenticated"

ReKey Summary

Station is Authenticated

↓

Authentication Server generates a Master Key

↓

Master Key encrypts Key Encryption Keys

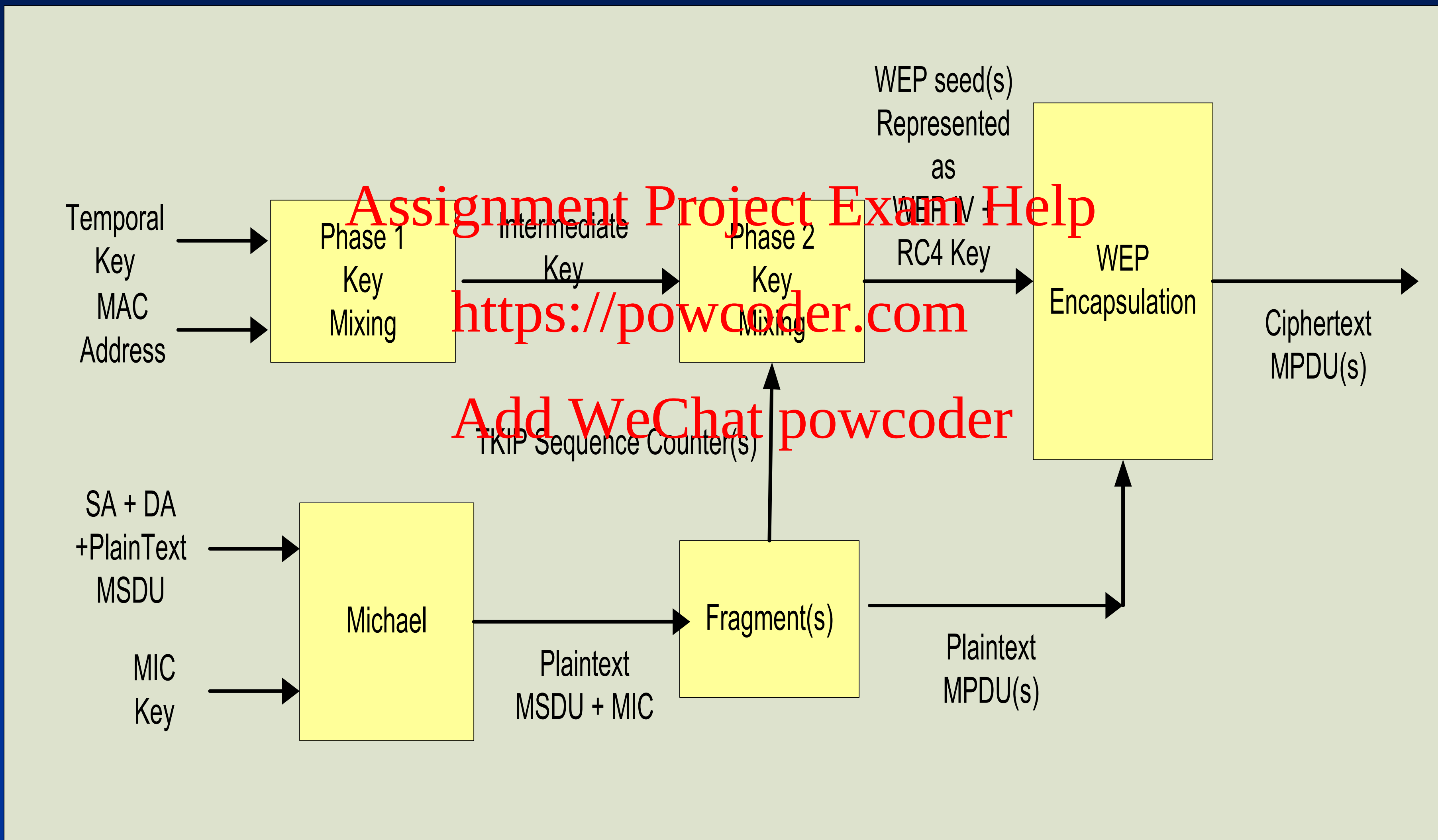
↓

Key Encryption Keys encrypt Temporal Keys

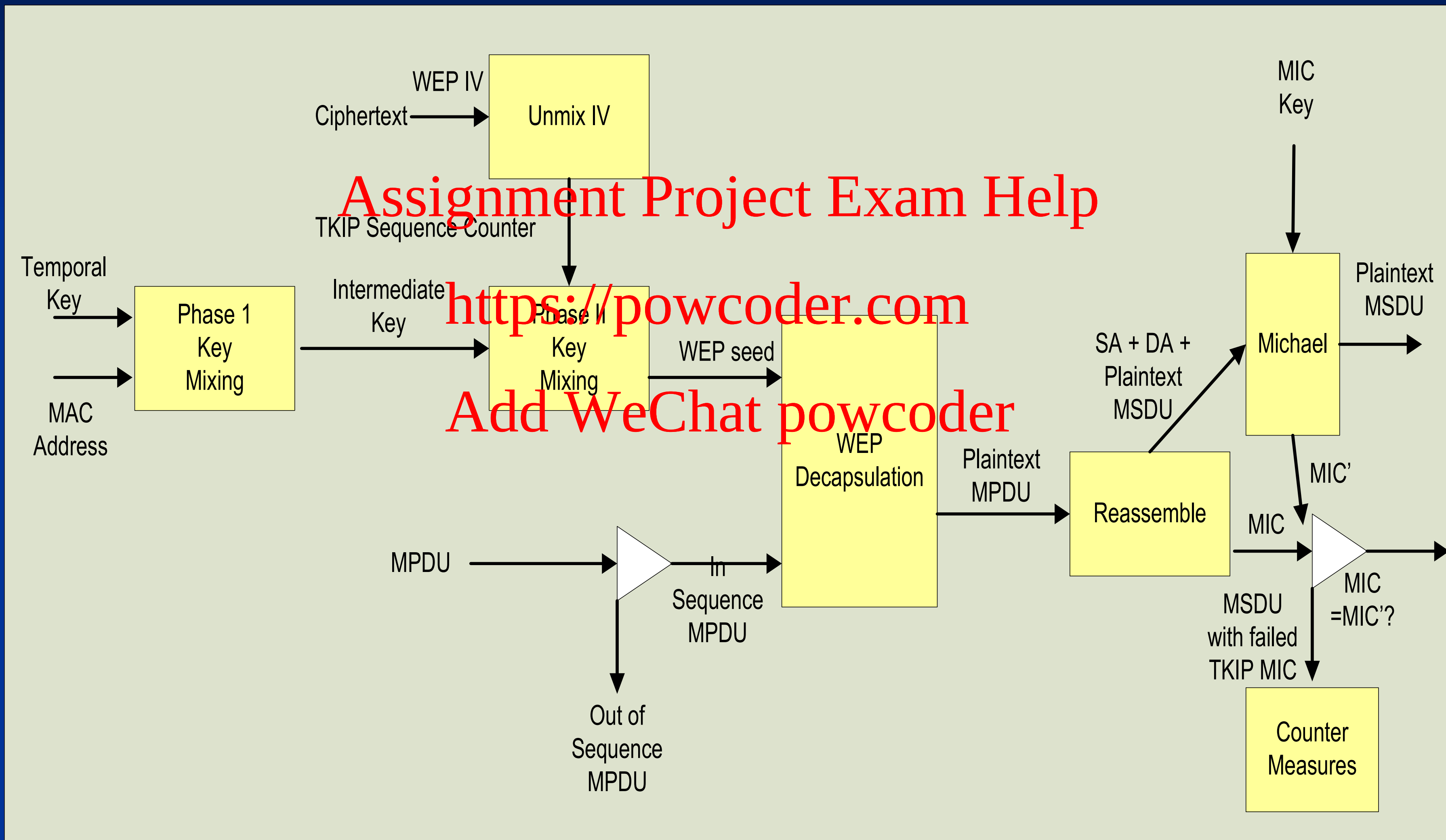
↓

Temporal Keys encrypt User Data

TKIP Encryption Process



TKIP Decryption Process



Q&A

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

References

- http://www.tech-faq.com/wireless-networks/tkip-temporal-key-integrity-protocol.shtml
- http://www.tech-faq.com/wireless-networks/tkip-temporal-key-integrity-protocol.shtml
- http://cache-www.intel.com/cd/00/00/01/77/17769_80211_part2.pdf