Georgia Tech

EFFICIENT CRYPTO TECHNIQUES FOR THE EDGE

MPC,ZK AND AUTHENTICATION

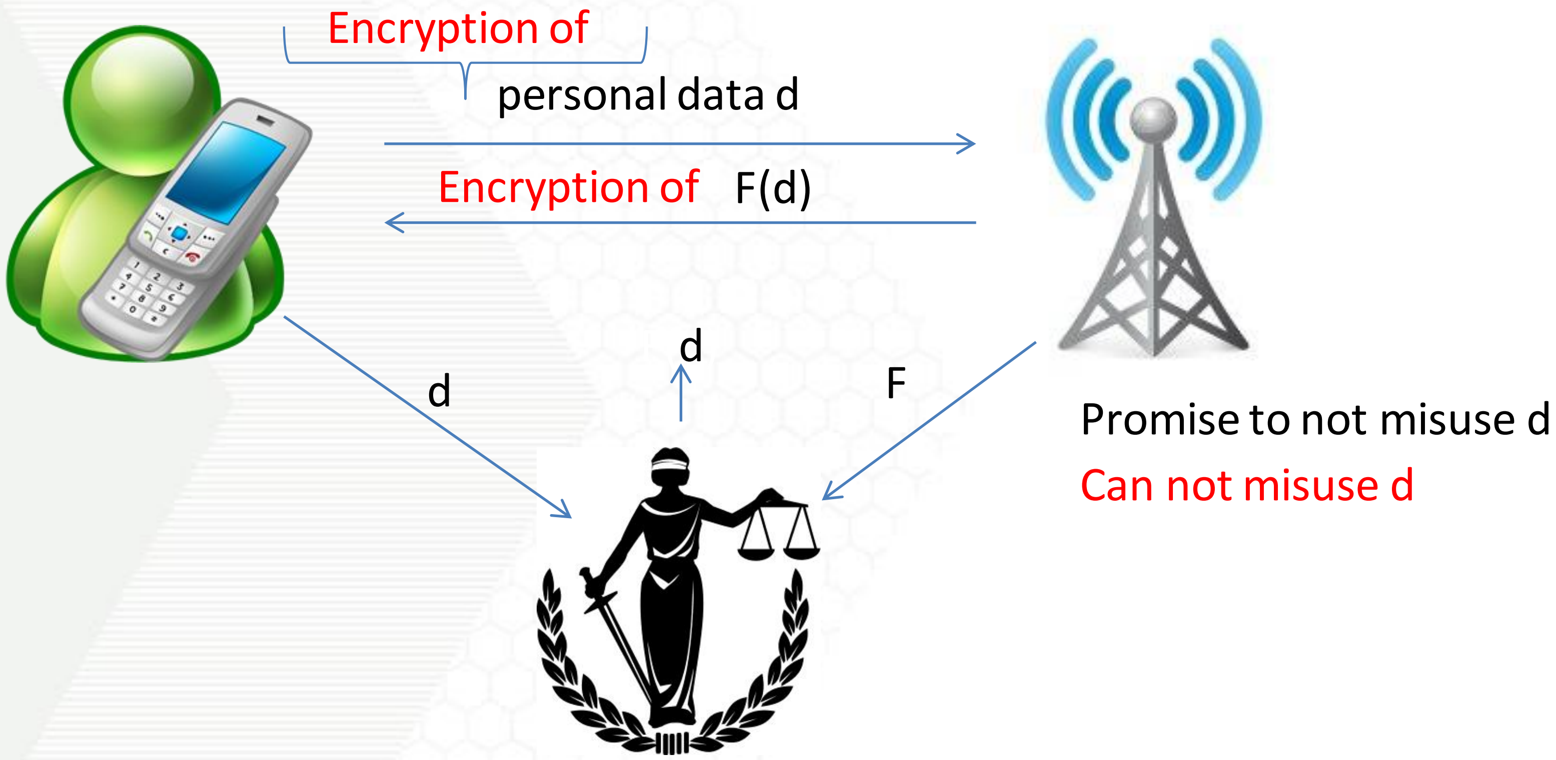
VLAD KOLESNIKOV (GTECH)

CREATING THE NEXT®

TRADITIONAL CRYPTOGRAPHY

- Encryption
  - (No one can learn our data)
  - Store and send files in secret.
- Authentication
  - (No one can fake our data)
  - Store and send files tamper proof.
- Enc + Auth => Secure communication
  - maybe

Georgia Tech

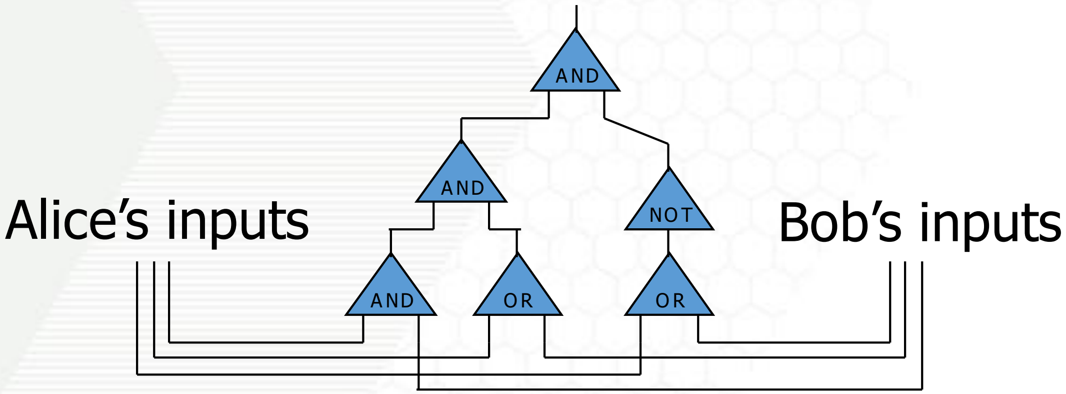


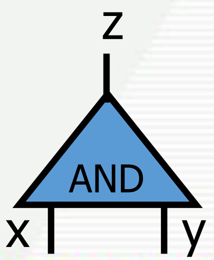
Any task involving a Trusted Third Party can also be implemented using a cryptographic protocol **without any loss of security**."

GARBLED CIRCUIT

Compute any function securely

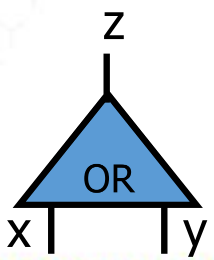
First, convert the function into a boolean circuit





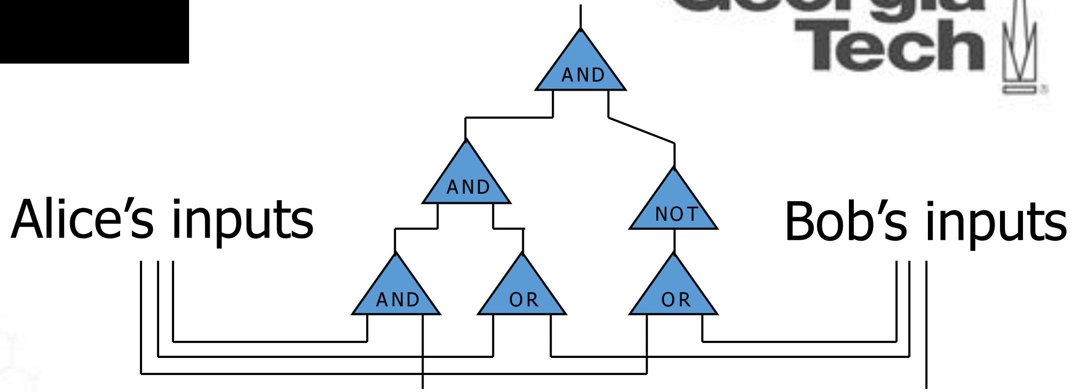
Truth table:

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1



Truth table:

x	y	z
0	0	0
0	1	1
1	0	1
1	1	1



Overview:

1. Alice prepares "garbled" version **C'** of C
2. Sends "encrypted" form **x'** of her input x
3. Allows Bob to obtain "encrypted" form **y'** of his input y
4. **Bob can compute from C',x',y' the "encryption" z' of z=C(x,y)**

   Think "Evaluation under encryption"
5. Bob sends **z'** to Alice and she decrypts and reveals to him z

Crucial properties:

1. Bob never sees Alice's input x in unencrypted form.
2. Bob can obtain encryption of y without Alice learning y.
3. Neither party learns intermediate values.
4. Remains secure even if parties try to cheat.

Cost:

2-5M Boolean gates/sec on 1Gbps LAN

AES function needs 6K Boolean gates.

ZERO-KNOWLEDGE (ZK) PROOFS

ZK-proof of

~~I am Alice;~~ I am authorized for resource R →

← Access to R

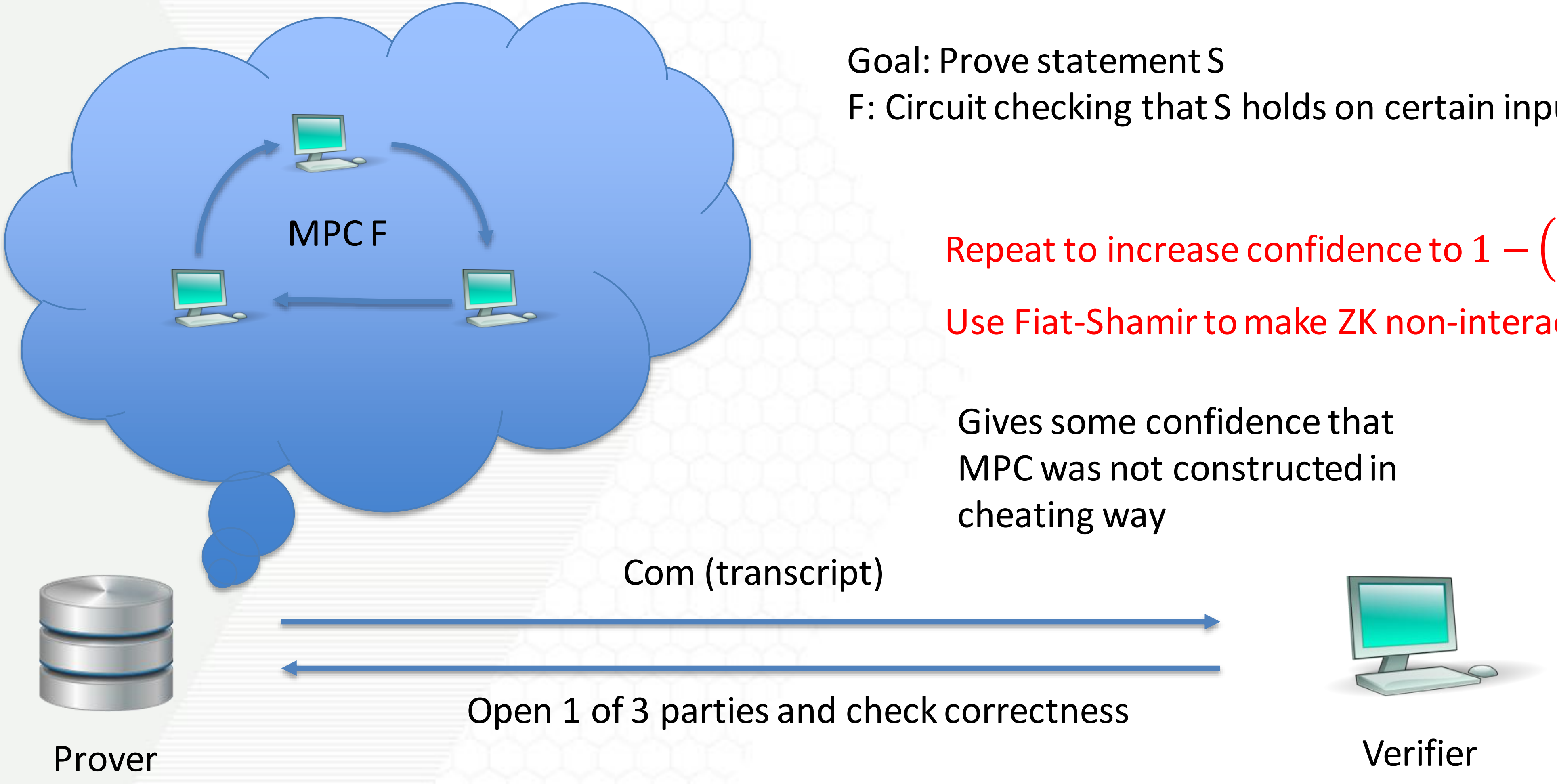
Promise to not misuse PII/use pattern

Can not misuse

ZK: proof that yields nothing but its validity; exist for any statement in NP

ZERO-KNOWLEDGE FROM MPC-IN-THE-HEAD [KATZ,K,WANG, CCS 2018]

[IKOS07]:



Goal: Prove statement S

F: Circuit checking that S holds on certain inputs

Repeat to increase confidence to $1 - \left(\frac{2}{3}\right)^k$

Use Fiat-Shamir to make ZK non-interactive

Gives some confidence that MPC was not constructed in cheating way

Cost: several KB and several ms. E.g. 45KB/30ms for signature generation

MESSAGE AUTHENTICATION CODE (MAC)

- Symmetric key equivalent of public key signatures
  - Need secret key to sign and to verify
  - Cannot sign without secret key

Deterministic MAC: $\{0,1\}^n \times \{0,1\}^n \rightarrow \text{TAG}$

Verification: recompute and compare the tag

Security:

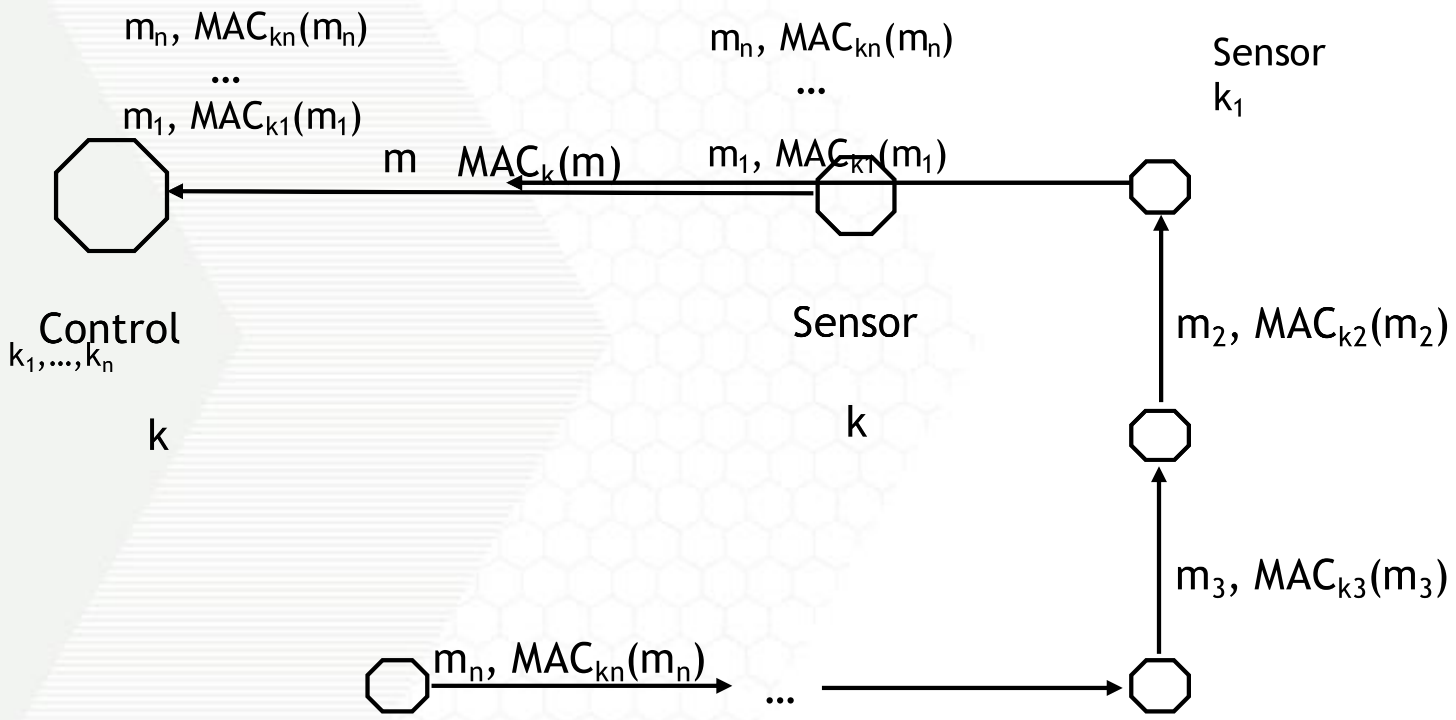
$\text{k} \in_R \{0,1\}^n$, Adv can access oracle $\text{O(m)} = \text{MAC}_k\text{(m)}$

Adv cannot forge MAC:

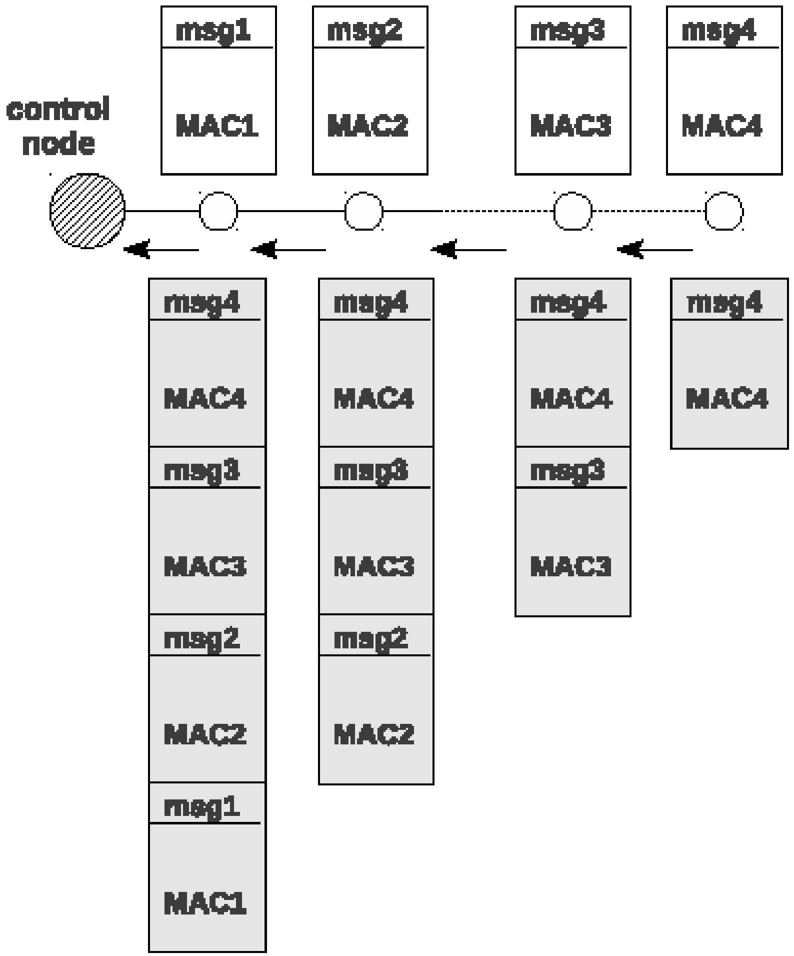
Output (m', t'), where he never queried O(m')

MAC AGGREGATION

Authentication of sensor data – multiple sensors, hop-by-hop network



MAC AGGREGATION

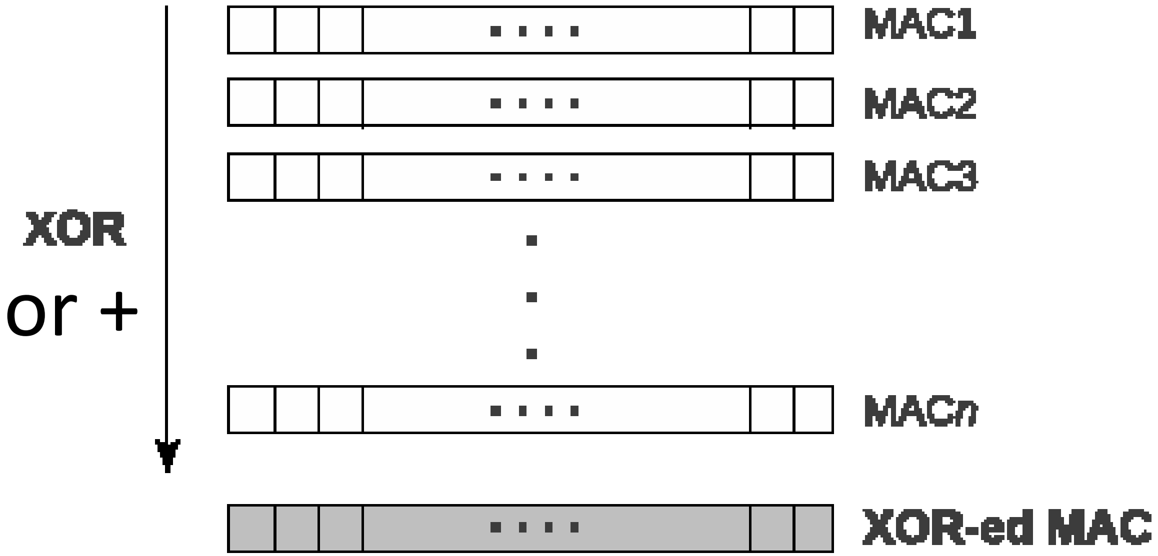


Issue: Authenticators may overwhelm payload traffic

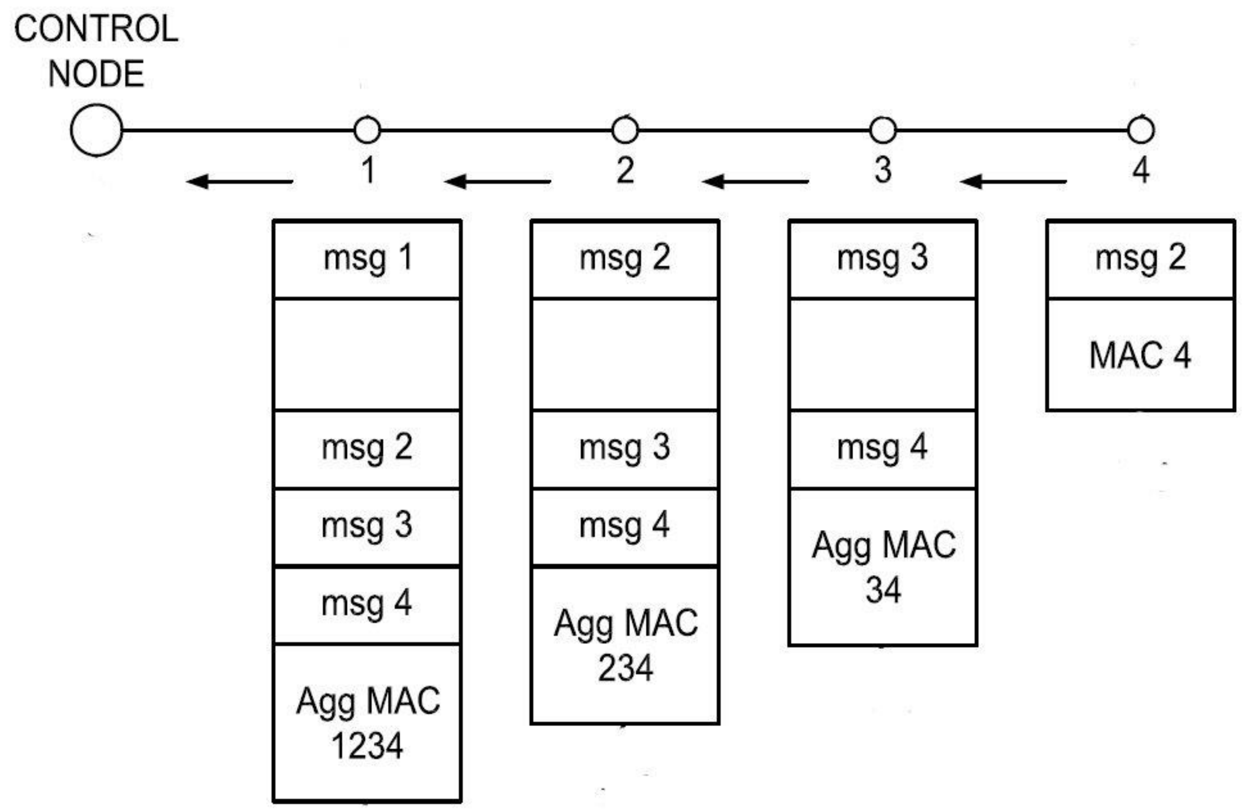
AGGREGATED MAC (KATZ-LINDELL08, K12)

Application of signature aggregation into PSK domain

Very simple idea:



AGGREGATED MAC (KATZ-LINDELL08, K12)



Theorem [KL08,K12]: If MAC1, …, MACn are unforgeable, then XOR-MAC and +MAC is also unforgeable.