

# COMP32412: Internet of Things Architecture and Applications

Countermeasures and techniques to secure IoT

**Mustafa A. Mustafa**

[mustafa.mustafa@manchester.ac.uk](mailto:mustafa.mustafa@manchester.ac.uk)  
KB 2.93, 2nd Floor, Kilburn Building

## Overview & Learning Outcomes

- Describe and identify countermeasures to secure IoT systems
- Apply different techniques to secure IoT systems
- Describe and identify privacy-enhancing technologies

## Security Requirements of IoT



CONFIDENTIALITY



INTEGRITY



AVAILABILITY



AUTHENTICITY



PRIVACY

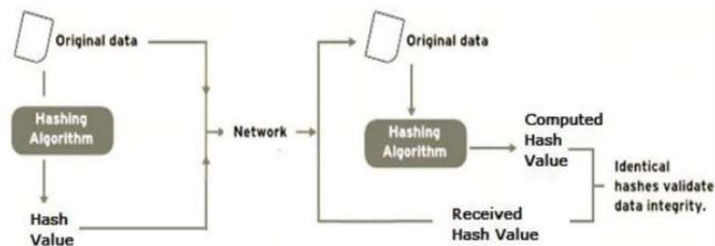
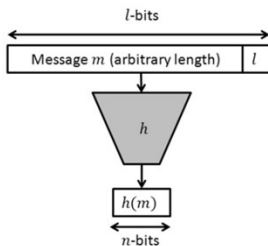


TRUST

## Integrity, authenticity and authentication

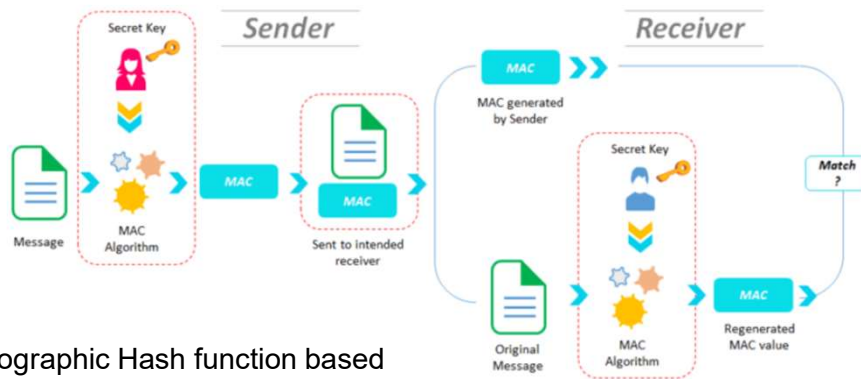
- **Integrity** ensures that the data can not be modified through illegal means and that the authorized users receive accurate data.
- **Authenticity** ensures that the data has not been modified and that the data has been originated by a specific user.
- **Authentication** ensures that only valid and authorised devices/users gain access to data and resources.
- These requirements are interrelated, and often simultaneously achieved by using one or more techniques.
- **Cryptographic hash functions, message authentication codes and digital signatures** are commonly used techniques.

## Cryptographic hash functions



- MD5 (**insecure**)
  - SHA1 (**insecure**)
  - RIPEMD-160
  - SHA2
  - SHA3
- It does not provide any assurance about originality.
  - The attacker can change the data and compute all together new hash and send to the receiver.
  - This integrity check application is useful only if the user is sure about the originality of data.

## Message authentication codes



Cryptographic Hash function based

- HMAC

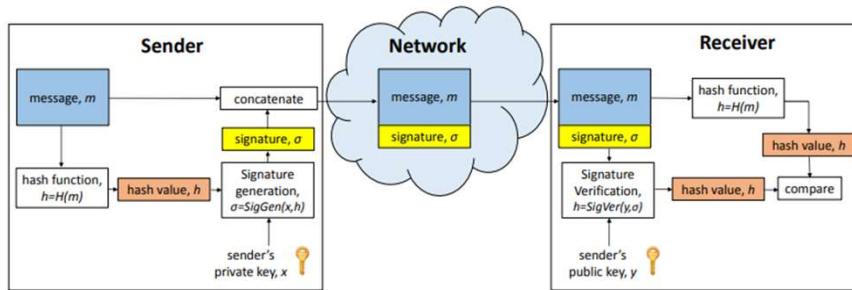
Block-cipher based

- OMAC, CCM, GCM

- It requires shared secret key.
- It does not provide non-repudiation.

- HMAC – Keyed-Hash message authentication code
- OMAC – One-time message authentication code
- CCM - counter with cipher block chaining message authentication code
- GCM - Galois/Counter Mode

## Digital signatures



- RSA
  - DSA
  - ECDSA
  - BLS
- It is based on public/private cryptography.
  - In addition to integrity and data authenticity, it provides non-repudiation too.

- RSA – Rivest–Shamir–Adleman
- DSA – Digital Signature Algorithm
- ECDSA – Elliptic Curve Digital Signature Algorithm
- BLS – Boneh–Lynn–Shacham short signatures

## Example use: Automated Software update

### Server

- Hashes the update
- Generates signature of the hash value
- Sends update + signature of the hash value

### IoT device

- Hashes the received update
- Verifies the signature of the hash value
- Compares the generated and received hash value

### Server

- Hashes the update
- Publishes the hash value at a secure server
- Sends update

### IoT device

- Hashes the received update
- Compares the generated and published hash value



## Security Requirements of IoT



CONFIDENTIALITY



INTEGRITY



AVAILABILITY



AUTHENTICITY



PRIVACY

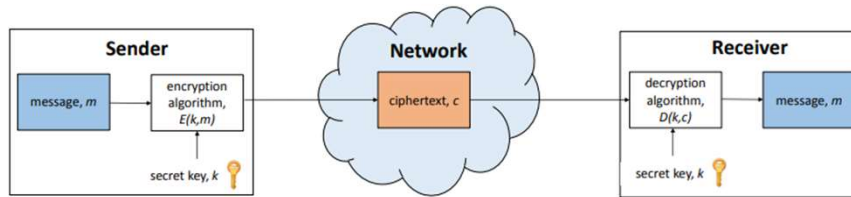


TRUST

## Confidentiality

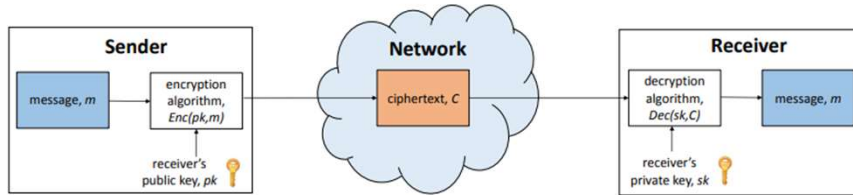
- **Confidentiality** ensures that the system data is only available to the authorised users and no other user can read the data.
- **End-to-end encryption** is the most commonly used technique.
- **Symmetric key** and **Public key cryptography** are the two main branches of the encryption technique.

## Symmetric key cryptography



- DES (**insecure**)
- 3DES (**insecure**)
- AES (Cryptography Standard)
- Ascon (**Lightweight Cryptography Standard**)
- Speck (**Controversial scheme – designed by NSA and withdrawn by NIST**)
- Efficient
- Quantum computing resistant
- Require shared secret key

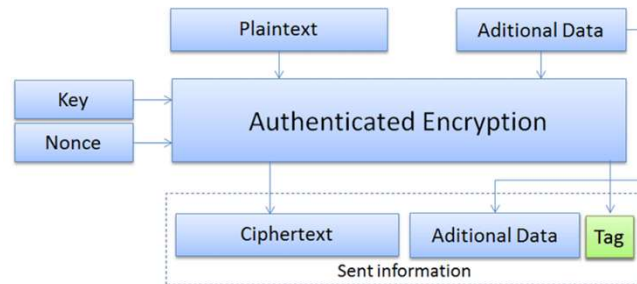
## Public key cryptography



- RSA
- ElGamal
- DH
- ECDH
- Less efficient
- Not quantum computing secure
- It does not require shared secret key

- DH – Diffie–Hellman
- ECDH - Elliptic-curve Diffie–Hellman

## Integrity + authenticity + confidentiality



- **Authenticated encryption (AE)** - an encryption scheme which simultaneously assures the data confidentiality and authenticity
- **Authenticated encryption with Associated Data (AEAD)** - allows "associated data" (AD) which is not made confidential, but its integrity is protected.
- Typical AD is the header of a network packet that contains its destination.

## Security Requirements of IoT



CONFIDENTIALITY



INTEGRITY



AVAILABILITY



AUTHENTICITY



PRIVACY



TRUST





## Privacy

- **Privacy** ensures that the data can only be controlled by its corresponding user only and no other user can access or process the data.
- Typical privacy-enhancing techniques are:
  - Perturbation
  - Secure computation techniques
    - Homomorphic encryption
    - Multiparty computation
  - Anonymous communication
    - Tor
    - Mix-nets

## Perturbation – Zero-sum masking

Zero-sum masking aggregation technique uses random numbers which when added together sum to zero

### IoT devices

	m1	r1
	m2	r2
	m3	r3
	m4	r4

$$r1+r2+r3+r4 = 0$$

### Data sent

m1+r1  
 m2+r2  
 m3+r3  
 m4+r4

### Server side



$$\text{sum}(m_i+r_i) = \text{sum}(m_i)$$

- Uses only addition and subtraction operations, it is very efficient
- Only weak security guaranties can be achieved
- It does not tolerate data losses
- It often requires coordination between IoT devices

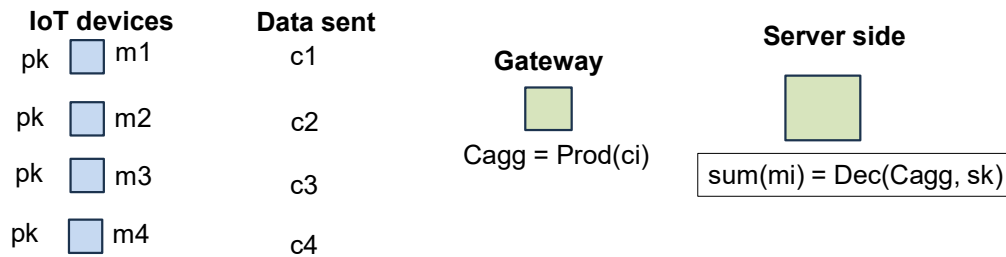
16

- As this technique only uses addition and subtraction operation it is very efficient.
- However, since no cryptographic techniques are used, they usually offer only weak security guaranties.
- This technique does not tolerate data losses, as even if one data is lost, the random numbers will not sum to zero.
- As at each time slot new random numbers needs to be generated, this technique often requires coordination between IoT devices.



## Homomorphic encryption

Homomorphic encryption is a specific type of an encryption scheme which allows operations to be performed directly on the ciphertexts.



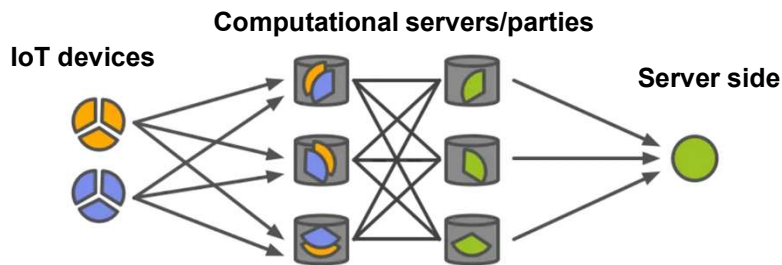
- It relies on cryptographic primitives; hence it provides strong security guaranties.
- It tolerates data losses
- No device coordination is required
- However, since it relies on cryptographic primitives, it is not very efficient.

17

- Homomorphic encryption relies on cryptographic primitives, which provide strong security guaranties.
- In addition, it also tolerates data losses as any lost data will not affect the aggregated result of the received data.
- Since no new cryptographic key is required at each time slot, no device coordination is required.
- However, since it relies on cryptographic primitives, it is computationally heavy – it requires encryption and decryption operations.

## Multiparty computation

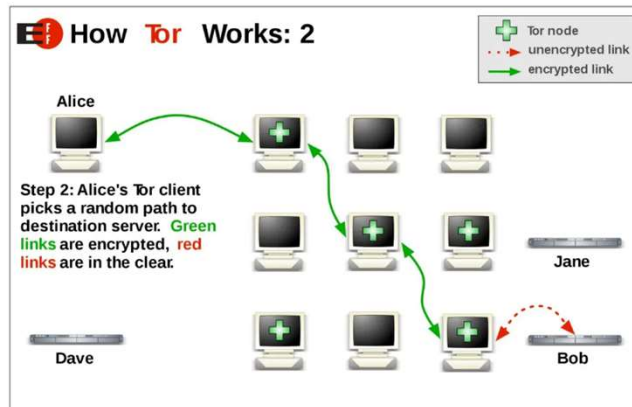
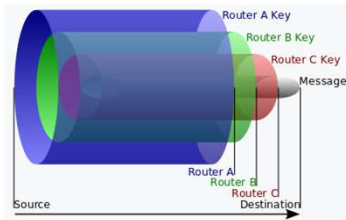
Multiparty computation allows two or more parties to do a computation together and get the results without disclosing any of the parties' sensitive input.



- It provides strong security guaranties.
- It tolerates data losses.
- No IoT device coordination is required.
- However, it adds considerable communication cost and requires multiple parties.

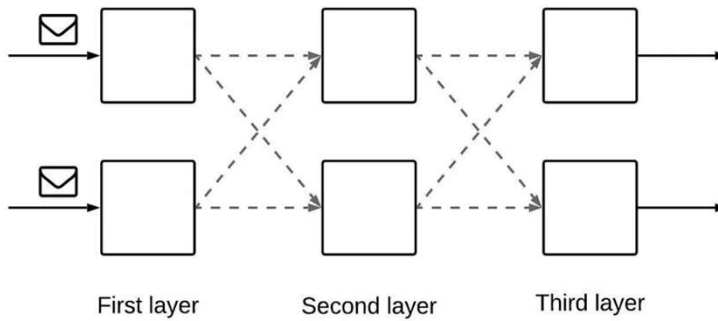
## Tor – onion routing

**Onion routing** is a technique for anonymous communication. In an **onion network**, messages are encapsulated in layers of encryption, analogous to the layers of an onion.



## Mix-nets

A mix network mixes packets to destroy information about the time they were sent, which can be used by attackers to determine who sent the packets and even what data they contain.



## Summary

- We have looked at different techniques for providing guarantees for various security and privacy requirements.
- Fulfilling all the security/privacy requirements **efficiently** is challenging.
- IoT devices often have limited computations, which requires more lightweight solutions.