Secure Sockets

Part 4 – TLS Handshake

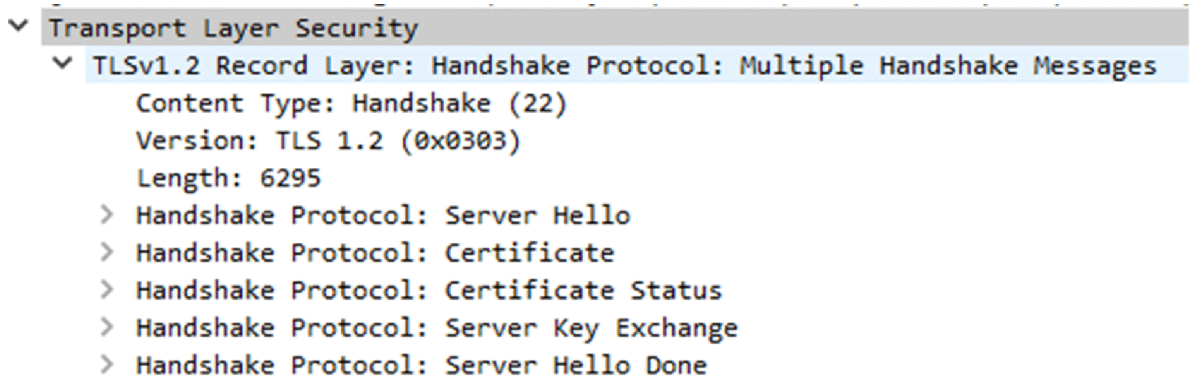
Barak Gonen

Part 4 – TLS Handshake

- Records
- RSA Handshake
- Diffie Helman Handshake
- Session Resumption
- Extensions
  - Server Name Indication
  - Session Tickets
  - OCSP Stapling
- TLS Decryption

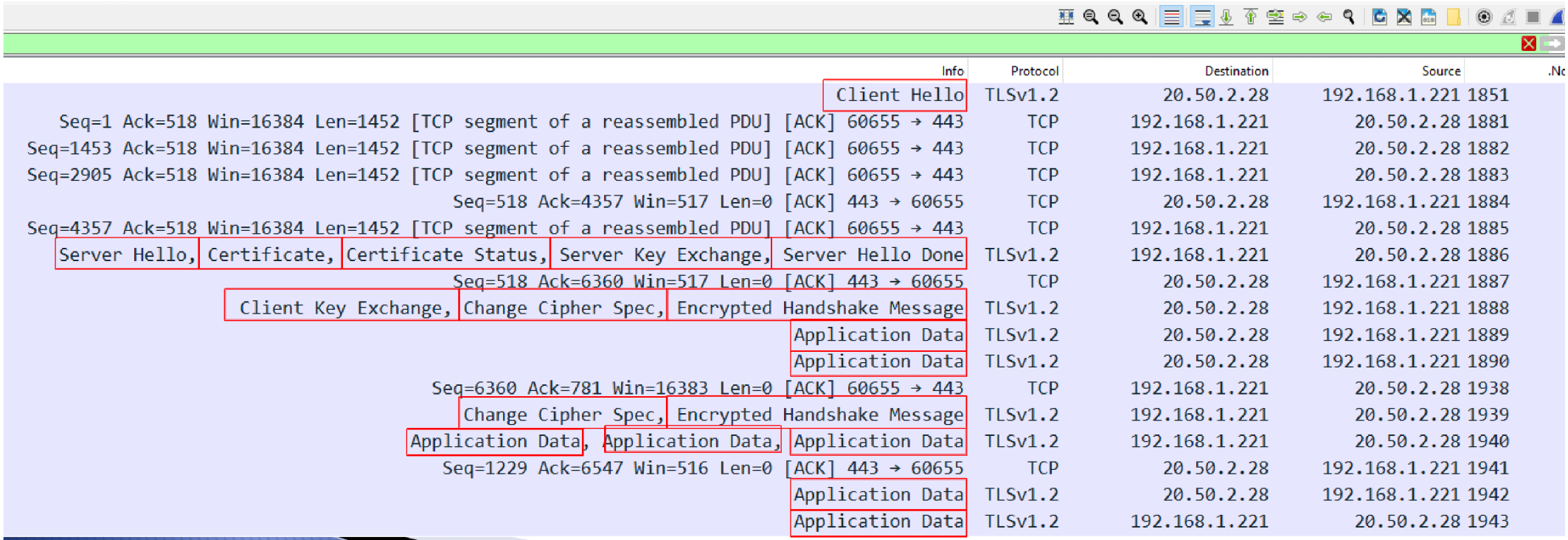
Record Layer Structure

- 1. Content type
  - handshake, application data etc.
- 2. Version
  - 0x0301 – TLS 1.0
  - 0x0302 – TLS 1.1
  - 0x0303 – TLS 1.2
  - 0x0304 – TLS 1.3
  - Sometimes fake (lower) to bypass middleboxes
- 3. Length – bytes
- 4. Records



Records

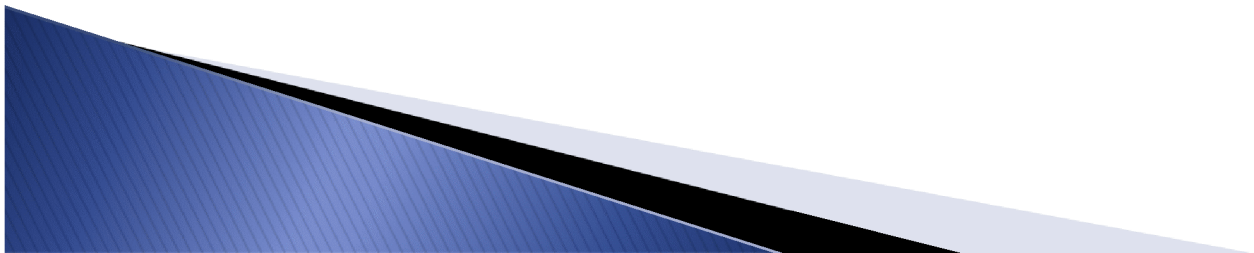
- Handshake – Client Hello, Server Hello
- Change Cipher Spec
- Application Data
- Alert



Info	Protocol	Destination	Source	.No
Client Hello	TLSv1.2	20.50.2.28	192.168.1.221	1851
Seq=1 Ack=518 Win=16384 Len=1452 [TCP segment of a reassembled PDU] [ACK] 60655 → 443	TCP	192.168.1.221	20.50.2.28	1881
Seq=1453 Ack=518 Win=16384 Len=1452 [TCP segment of a reassembled PDU] [ACK] 60655 → 443	TCP	192.168.1.221	20.50.2.28	1882
Seq=2905 Ack=518 Win=16384 Len=1452 [TCP segment of a reassembled PDU] [ACK] 60655 → 443	TCP	192.168.1.221	20.50.2.28	1883
Seq=518 Ack=4357 Win=517 Len=0 [ACK] 443 → 60655	TCP	20.50.2.28	192.168.1.221	1884
Seq=4357 Ack=518 Win=16384 Len=1452 [TCP segment of a reassembled PDU] [ACK] 60655 → 443	TCP	192.168.1.221	20.50.2.28	1885
Server Hello, Certificate, Certificate Status, Server Key Exchange, Server Hello Done	TLSv1.2	192.168.1.221	20.50.2.28	1886
Seq=518 Ack=6360 Win=517 Len=0 [ACK] 443 → 60655	TCP	20.50.2.28	192.168.1.221	1887
Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message	TLSv1.2	20.50.2.28	192.168.1.221	1888
Application Data	TLSv1.2	20.50.2.28	192.168.1.221	1889
Application Data	TLSv1.2	20.50.2.28	192.168.1.221	1890
Seq=6360 Ack=781 Win=16383 Len=0 [ACK] 60655 → 443	TCP	192.168.1.221	20.50.2.28	1938
Change Cipher Spec, Encrypted Handshake Message	TLSv1.2	192.168.1.221	20.50.2.28	1939
Application Data, Application Data, Application Data	TLSv1.2	192.168.1.221	20.50.2.28	1940
Seq=1229 Ack=6547 Win=516 Len=0 [ACK] 443 → 60655	TCP	20.50.2.28	192.168.1.221	1941
Application Data	TLSv1.2	20.50.2.28	192.168.1.221	1942
Application Data	TLSv1.2	192.168.1.221	20.50.2.28	1943

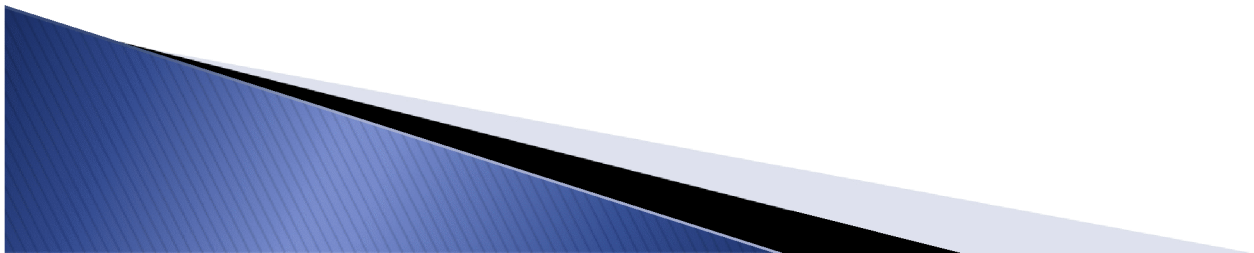
Record Types

- 20 Change Cipher Spec – start encrypting
- 21 Alert – Warning / Fatal
  - Handshake failed
  - Certificate expired
  - Etc.
- 22 Handshake
  - Sub-record types
- 23 Application Data



Exercise

- Create a Wireshark filter to filter only handshakes where the server chose RSA for key-exchange
- Solution:
  - tls.handshake.type == 2 and (tls.handshake.ciphersuite == 0x002f or tls.handshake.ciphersuite == 0x0035 or tls.handshake.ciphersuite == 0x009c or tls.handshake.ciphersuite == 0x009d)



TLS Handshake- RSA version

Client Hello

Server Hello

Certificate

Server Hello Done

Client Key Exchange

Change Cipher Spec

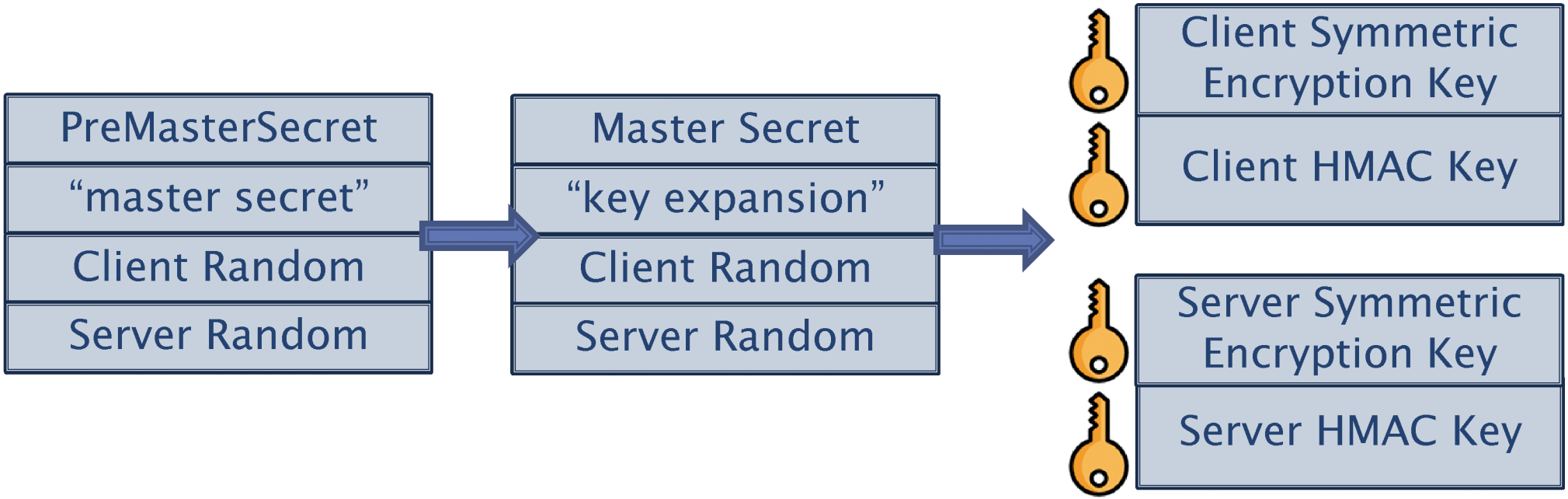
Client Finished

Change Cipher Spec

Server Finished

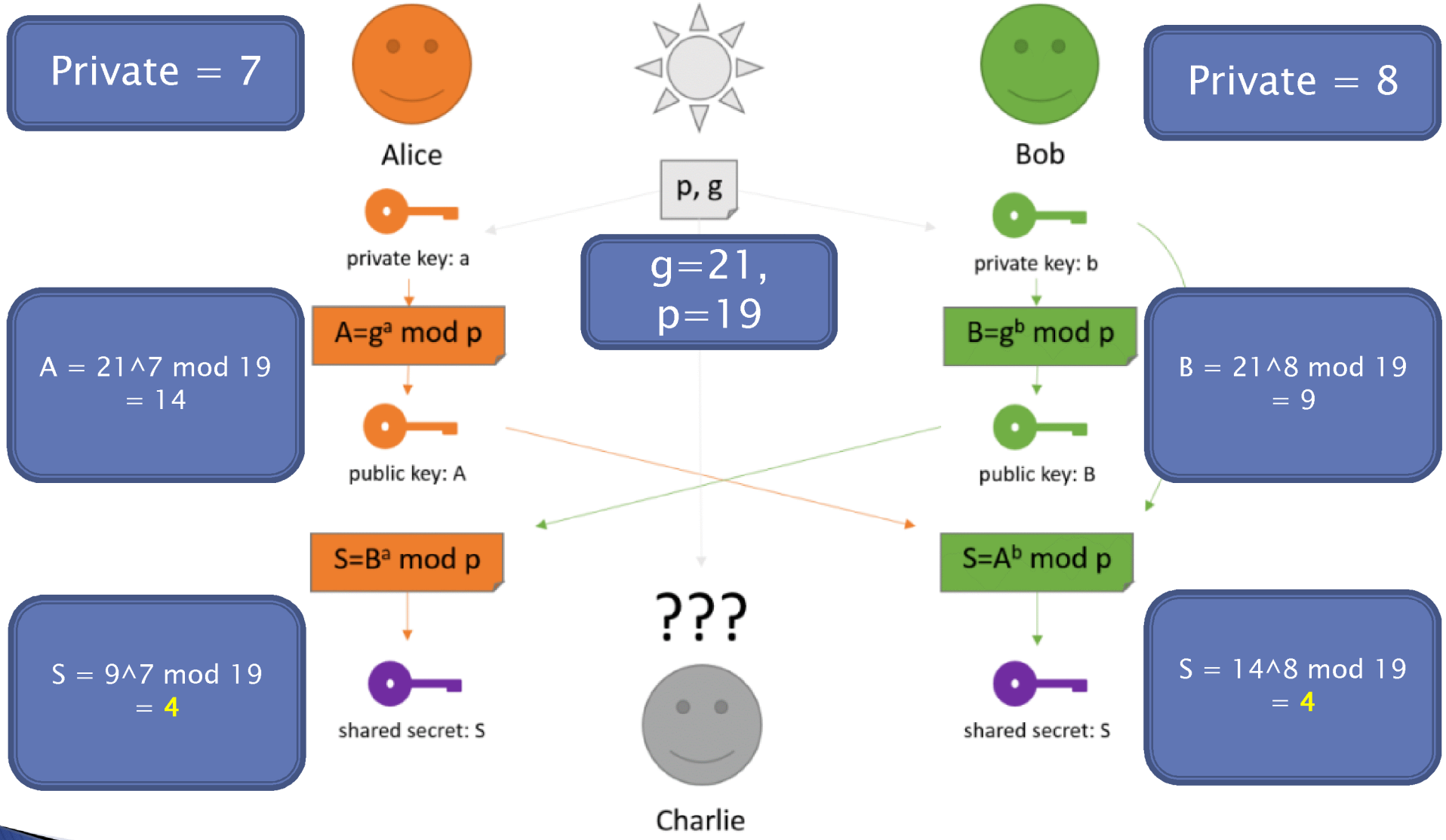
Client Key Exchange

- Random PreMasterSecret
  - Encrypted with Server's Public Key



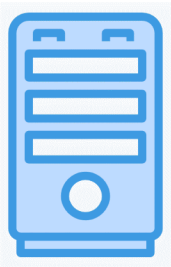
Handshake Finished

DH Algorithm



https://blog.noser.com/asymmetric-cryptography-diffie-hellman-key-exchange/

TLS Handshake- DH version



Client Hello

Server Hello

Certificate

Server Key Exchange

Server Hello Done

Client Key Exchange

Change Cipher Spec

Client Finished

Change Cipher Spec

Server Finished

Dive into Application Data

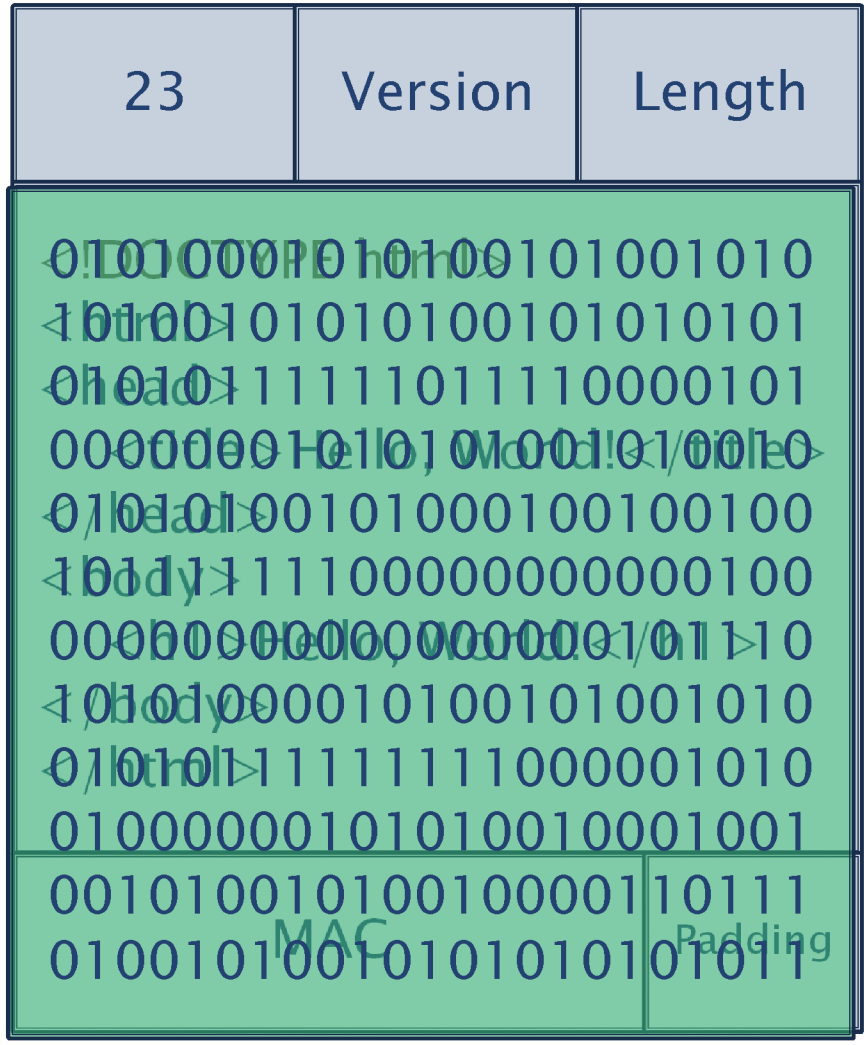
- C.I.A
  - Confidentiality – Symmetric Encryption
  - Integrity + Authentication – Message Auth. Code (MAC)

C.I.A Methods

- MAC-then-Encrypt
  - Known to have faults
- Encrypt-then-MAC
  - Suspected to have faults
- AEAD -Do both simultaneously

C.I.A Methods

- MAC-then-Encrypt
  - Known to have faults
- Encrypt-then-MAC
  - Suspected to have faults
- AEAD -Do both simultaneously



MAC-then-Encrypt scheme

AEAD

- MAC and Encrypt
- AEAD –
  - Header – MAC only
  - Data – Auth. + Enc.
- Used in:
  - Some TLS v1.2
  - TLS v1.3

2.1. Authenticated Encryption

The authenticated encryption operation has four inputs, each of which is an octet string:

A secret key K, which MUST be generated in a way that is uniformly random or pseudorandom.

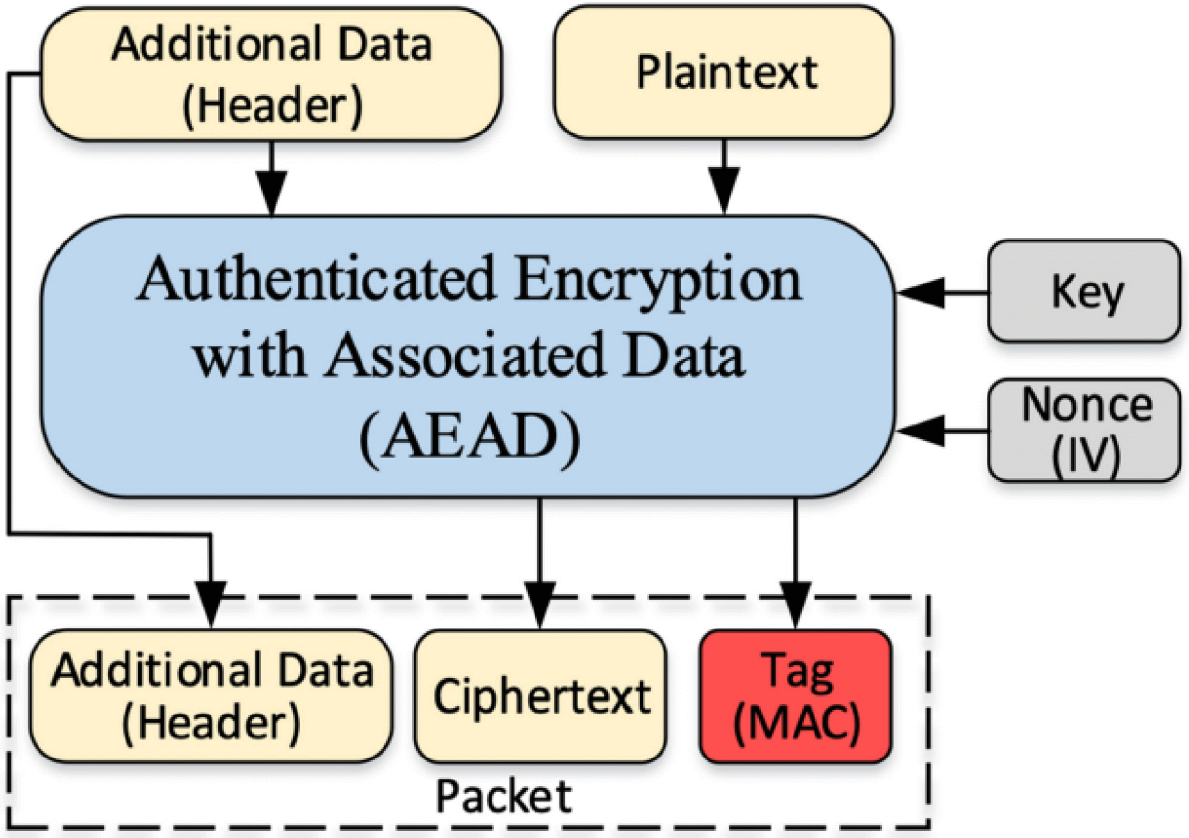
A nonce N. Each nonce provided to distinct invocations of the Authenticated Encryption operation MUST be distinct, for any particular value of the key, unless each and every nonce is zero-length. Applications that can generate distinct nonces SHOULD use the nonce formation method defined in Section 3.2, and MAY use any other method that meets the uniqueness requirement. Other applications SHOULD use zero-length nonces.

A plaintext P, which contains the data to be encrypted and authenticated.

The associated data A, which contains the data to be authenticated, but not encrypted.

RFC 5116

AEAD – Auth. Encryption with Associated Data



IEEE INTERNET OF THINGS JOURNAL, VOL. 7, NO. 1, JANUARY 2020

Application Data

- MAC and Encrypt
- AEAD –
  - Header – MAC only
  - Data – Auth. + Enc.
- Used in:
  - Some TLS v1.2
  - TLS v1.3