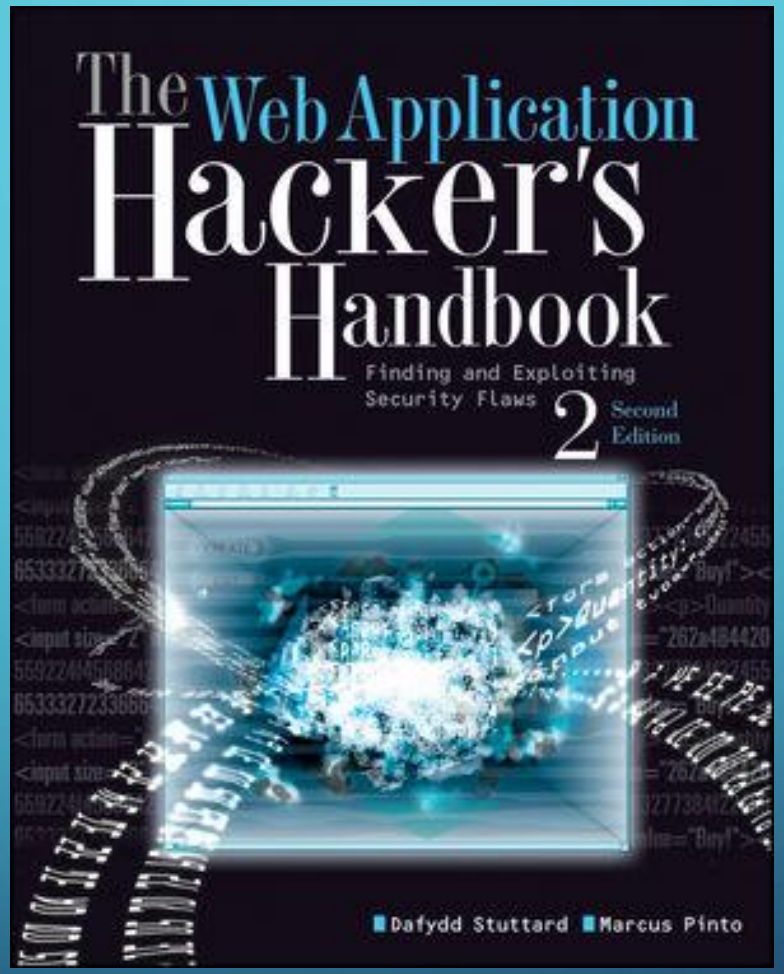
INTRO TO CRYPTOGRAPHY

HELLO!


The Web Application Hacker's Handbook
Finding and Exploiting Security Flaws
2 Second Edition
Dafydd Stuttard
Marcus Pinto

CRYPTOGRAPHY

GOALS OF CRYPTOGRAPHY

- Privacy
- Authentication
- Integrity
- Nonrepudiation

RANDOMNESS

System.Random

```
var r = new System.Random();
var randomValue = r.Next();

Console.WriteLine(randomValue);
```

System.Security.Cryptography.RandomNumberGenerator

```
using(var r = System.Security.Cryptography.RandomNumberGenerator.Create())
{
    var b = new Byte[4];
    r.GetBytes(b);

    Console.WriteLine(System.BitConverter.ToUInt32(b));
}
```

HASHING

HASHING

- Convert variable length data into fixed length values
- Useful for many applications in computing
- In security and cryptography we want one-way hashing functions
- Common hashes: MD5, SHA-1, SHA-512
- Common password hashes: Argon2, bcrypt, scrypt, PBKDF2
- Strictly speaking PBKDF2 isn't hashing but it'll be ok

ONE-WAY HASHING FUNCTIONS

PASSWORD HASHING

- Not all algorithms are appropriate for security usage!
- Primary goals of password hashing
  - Hard to brute force any individual password
  - Prevent attackers from attacking the entire DB at once
- Three attributes of good password hashing:
  - One-way function
  - Slow
  - Salted

.NET PASSWORD HASHING

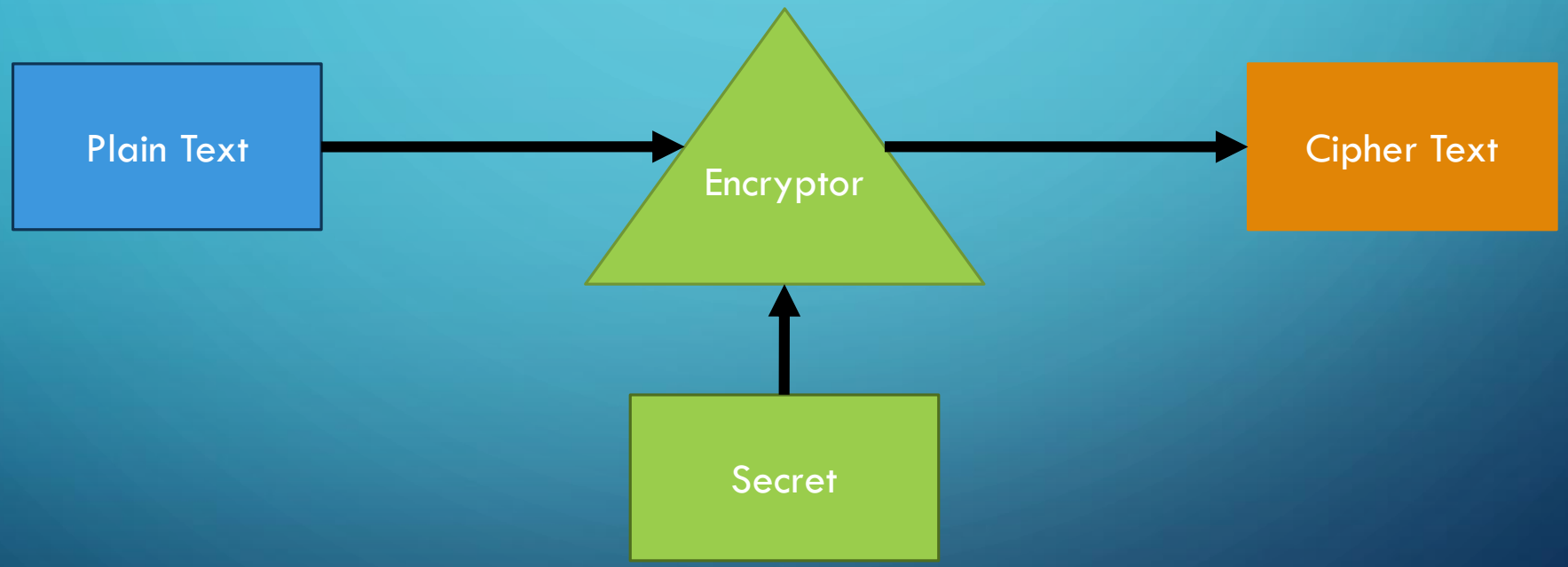
- Only PBKDF2 is available from Microsoft at this time
  - `System.Security.Cryptography.DeriveBytes.Rfc2898DeriveBytes`
    - The configurability of this implementation is lacking and the use of HMAC-SHA1 is not highly regarded today.
  - `Microsoft.AspNetCore.Cryptography.KeyDerivation.Pbkdf2`
    - `Allows for use of better underlying PRF such as HMAC-SHA512`
  - `Higher is better for iteration counts but if you have high traffic consider CPU load as well`

.NET PASSWORD HASHING

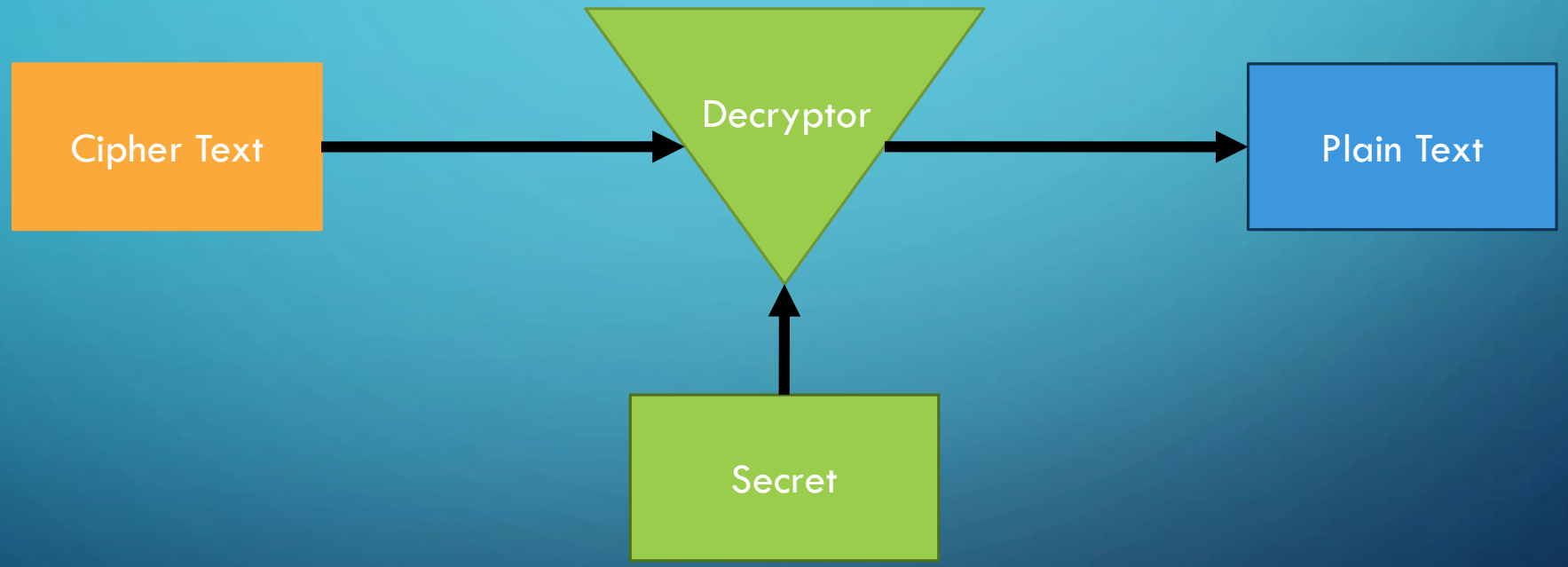
```
var badSalt = BitConverter.GetBytes(0xcbadcbadcbadcbad);
var password = "UltraSecure123!";
var iterations = 50000;

using(var k1 = new Rfc2898DeriveBytes(password, badSalt, iterations))
{
    Console.WriteLine(BitConverter.ToUInt64(k1.GetBytes(8)));
}
```

ENCRYPTION

ENCRYPTION



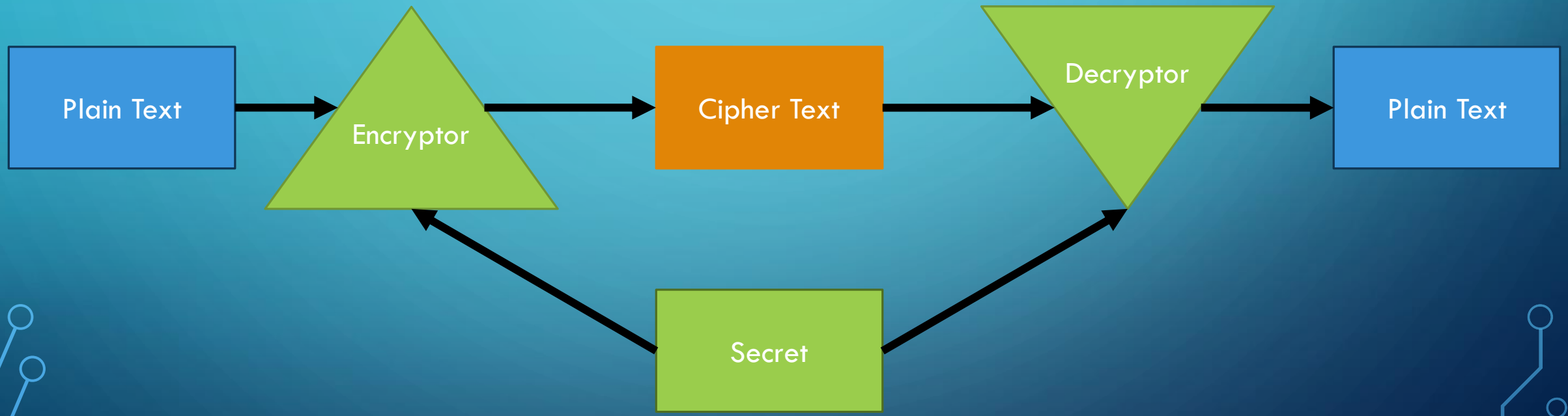
DECRYPTION



SYMMETRIC ENCRYPTION

- Processing is fast
- Key distribution is hard and high risk
- AES is a very common symmetric encryption algorithm in use

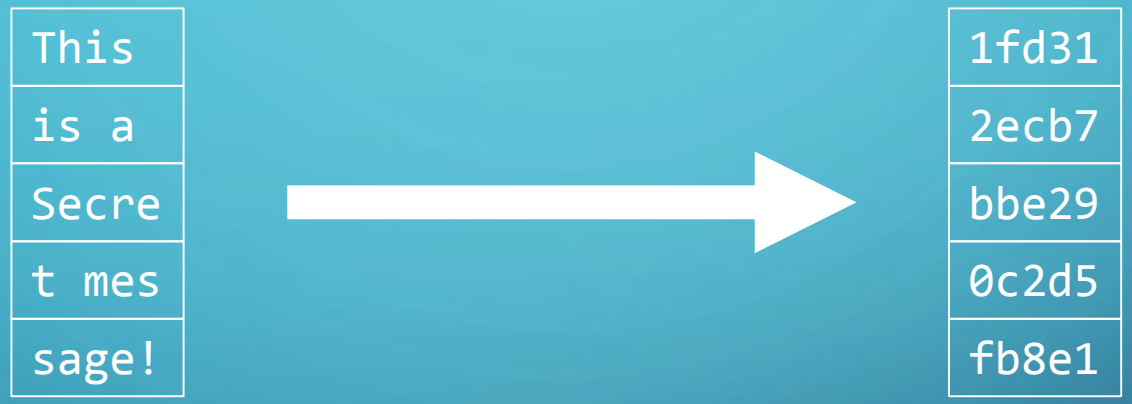
SYMMETRIC ENCRYPTION



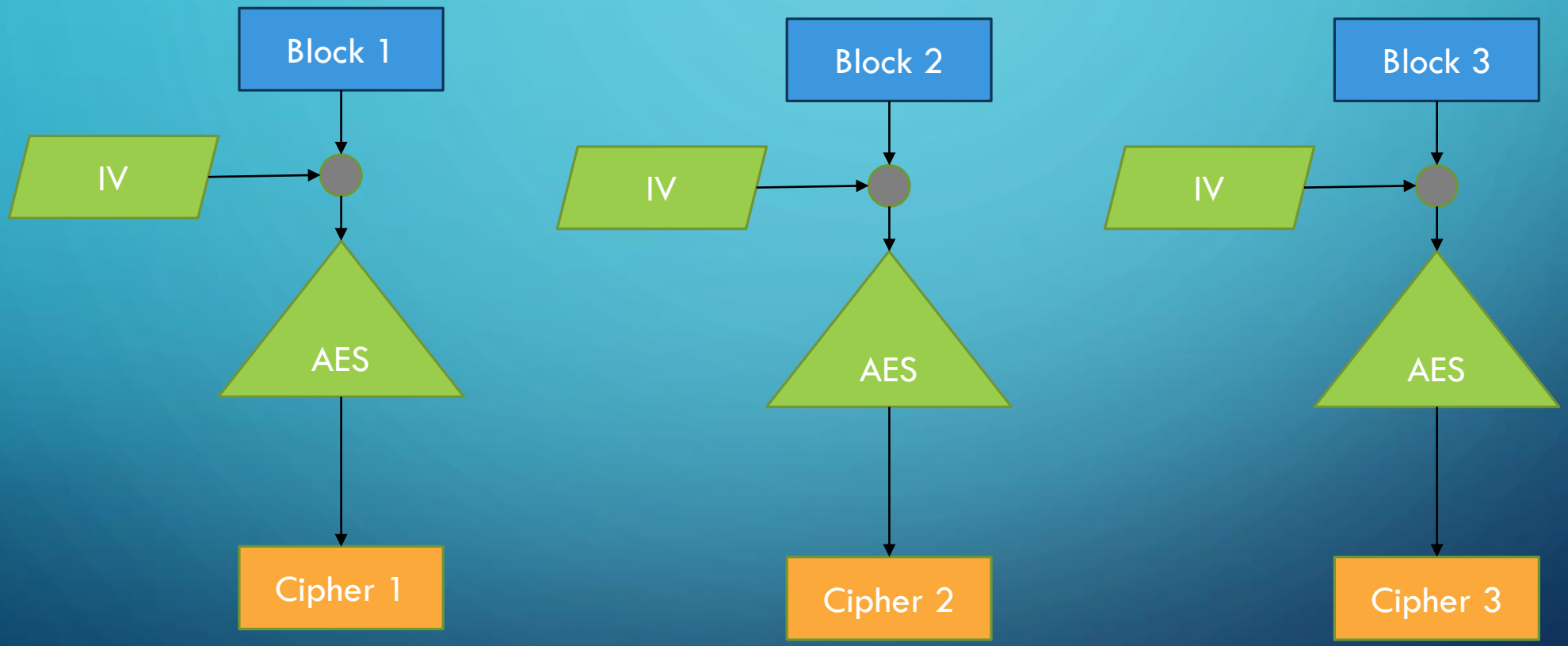
BLOCK ENCRYPTION

This is a secret message!

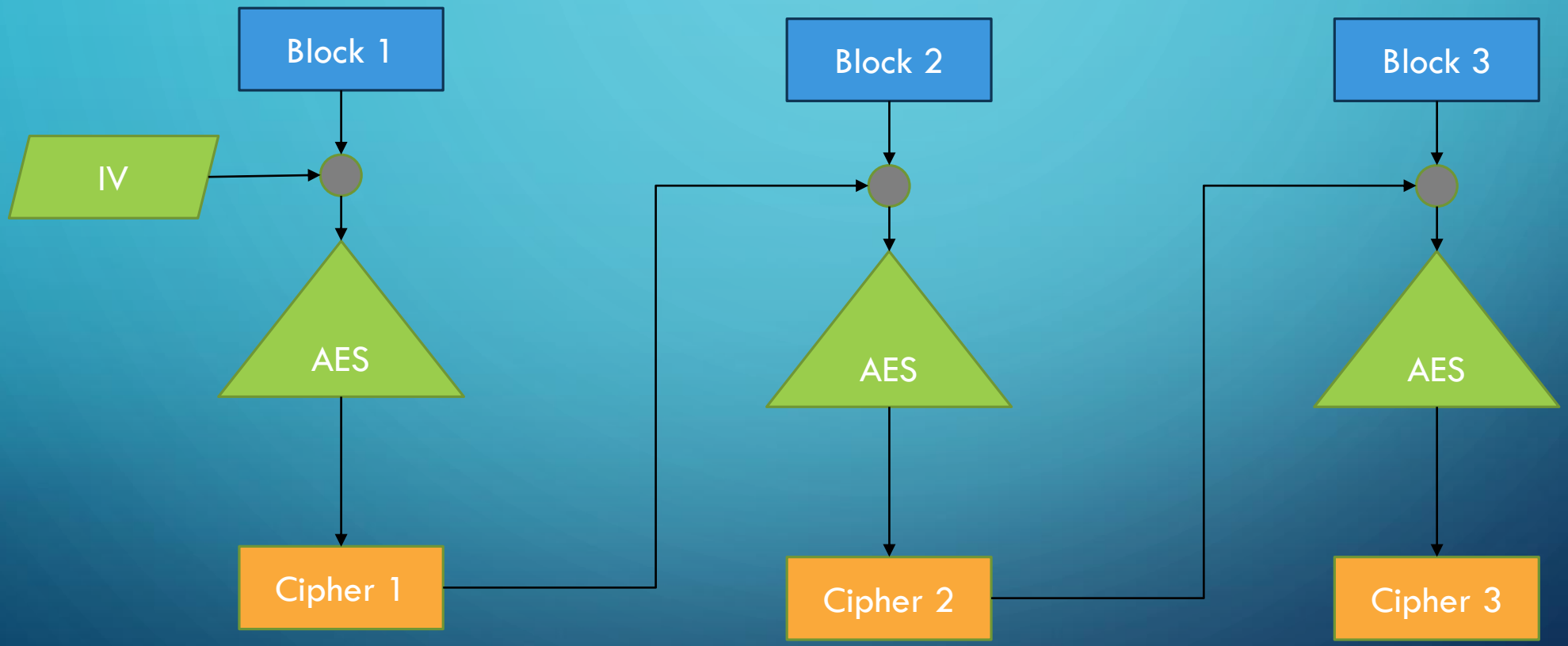
BLOCK ENCRYPTION

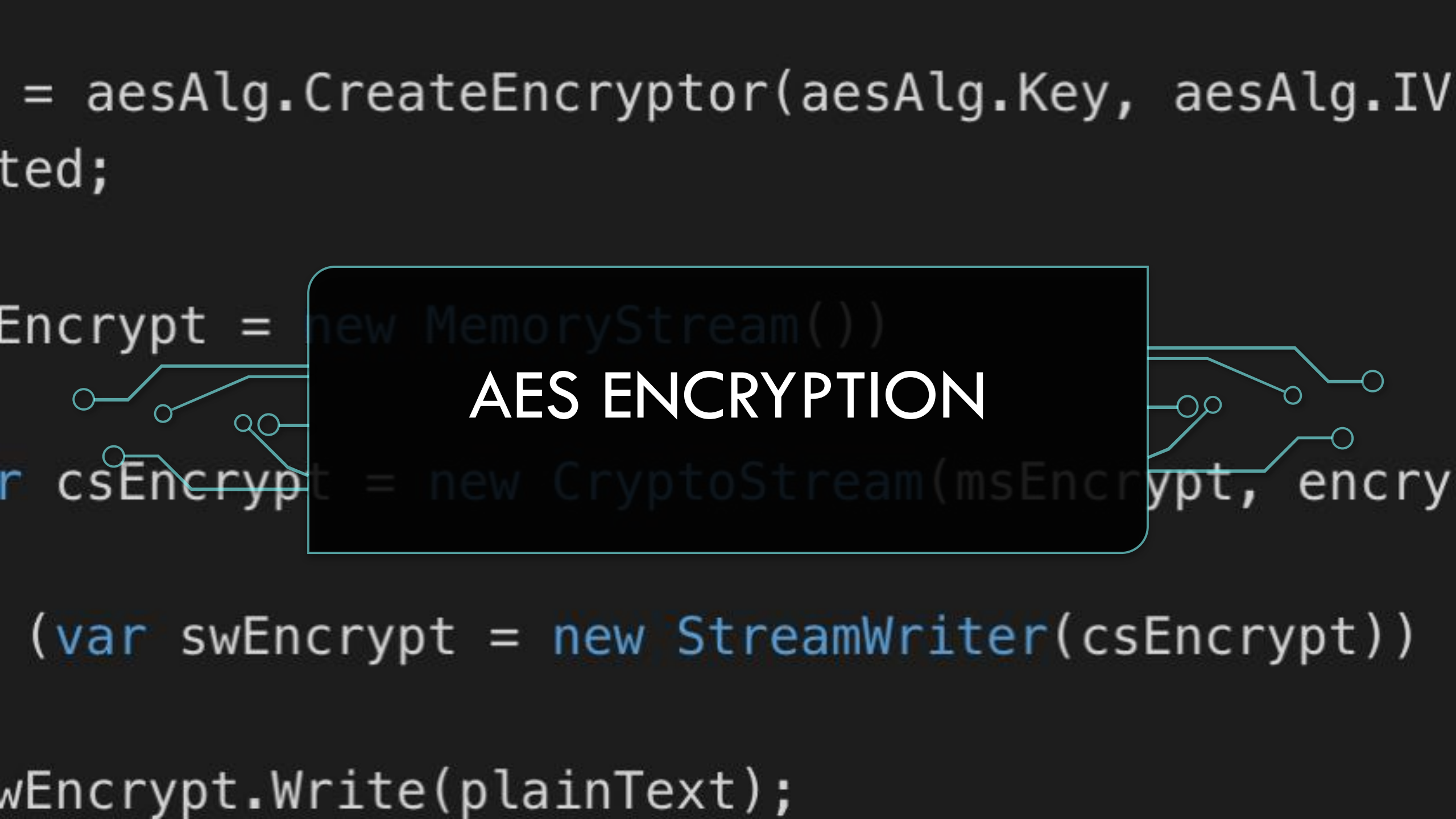


BLOCK ENCRYPTION - ECB



BLOCK ENCRYPTION - CBC

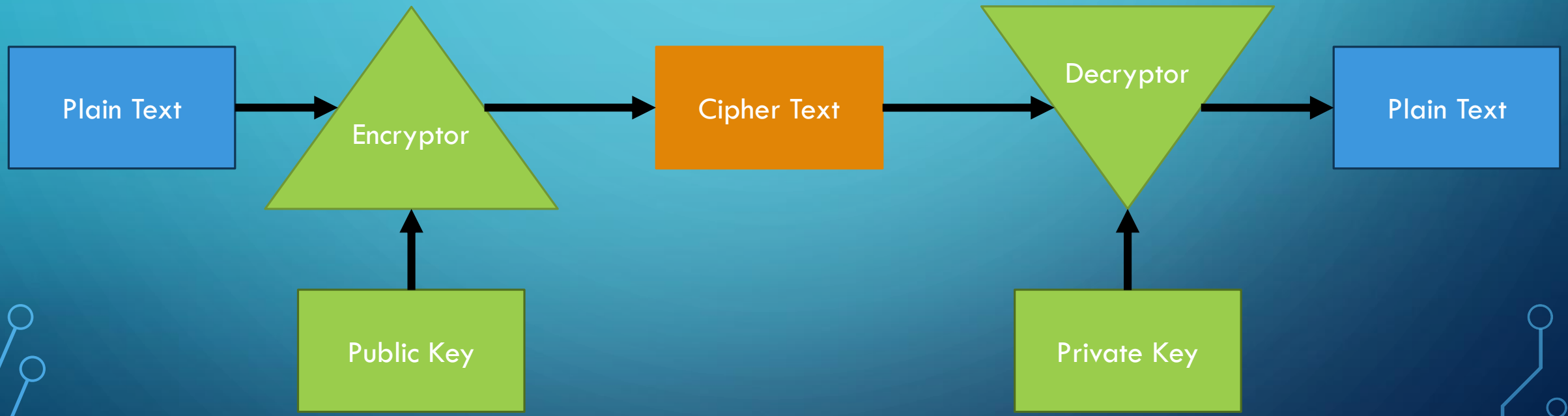



= aesAlg.CreateEncryptor(aesAlg.Key, aesAlg.IV
ted;
Encrypt =
r csEncrypt = new CryptoStream(msEncrypt, encry
(var swEncrypt = new StreamWriter(csEncrypt))
wEncrypt.Write(plainText);
AES ENCRYPTION

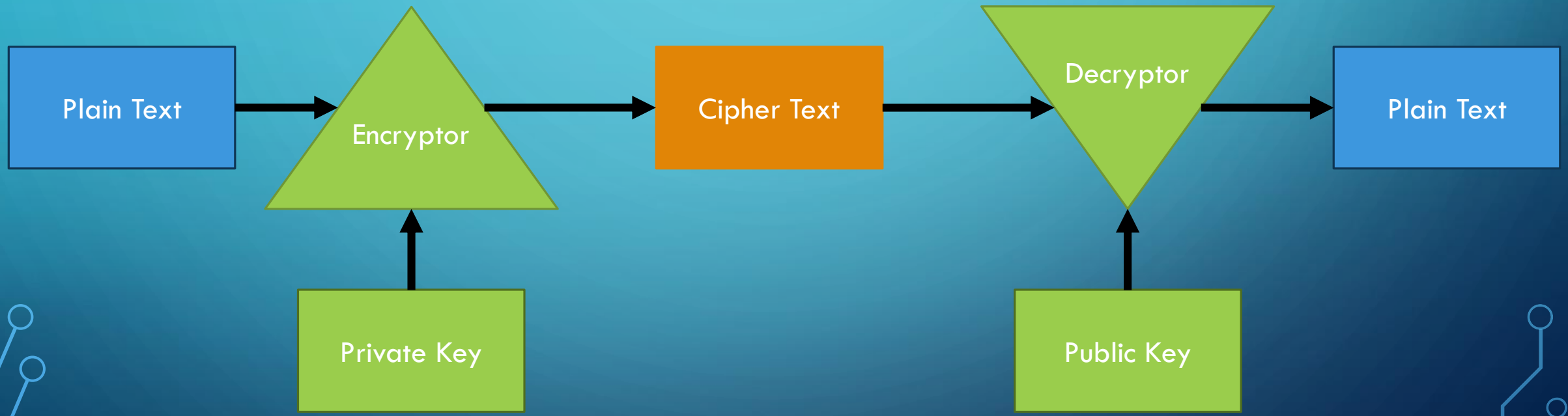
ASYMMETRIC ENCRYPTION

- Processing is slower
- Key distribution is easy
- RSA is a very common asymmetric encryption algorithm in use
- TLS and PGP are common applications of asymmetric encryption

ASYMMETRIC ENCRYPTION



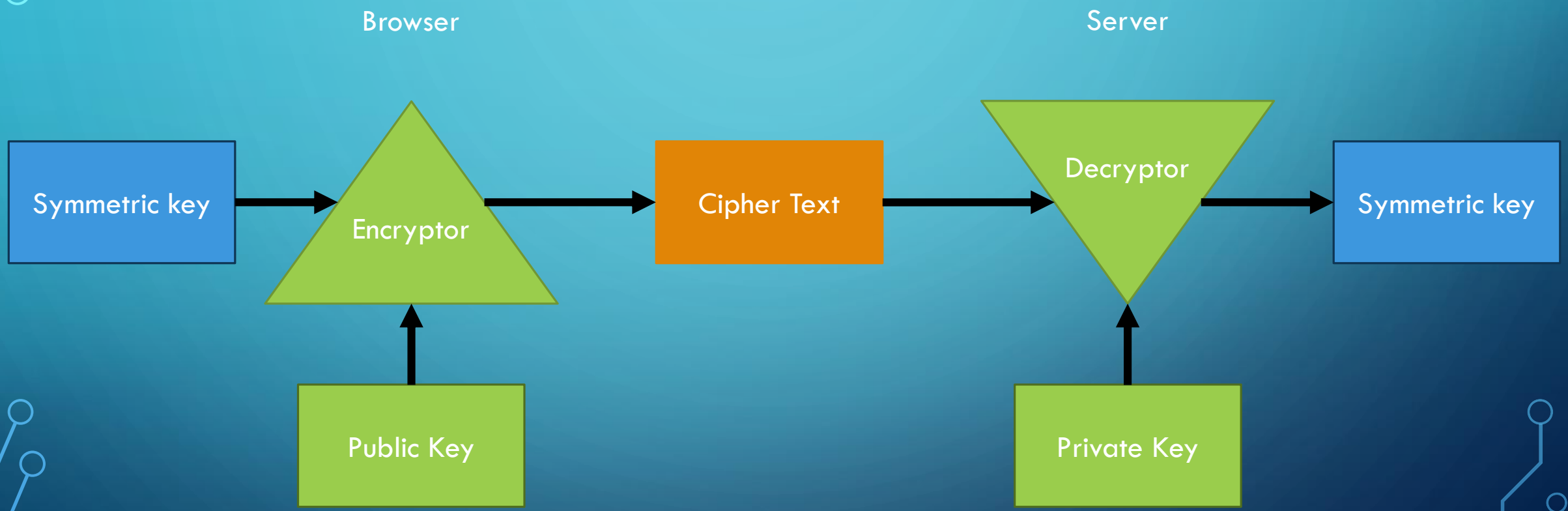
ASYMMETRIC ENCRYPTION



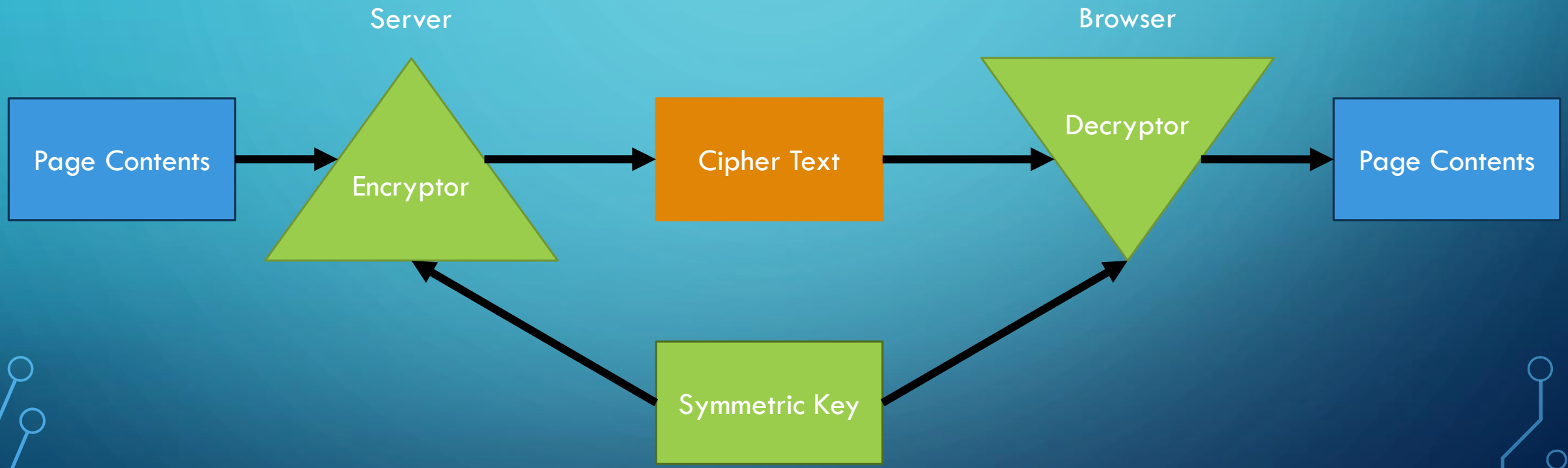
HYBRID ENCRYPTION

- TLS and PGP are hybrid mechanisms
- Use asymmetric encryption to share a key for symmetric encryption
- Remaining data is sent using symmetric encryption

HYBRID ENCRYPTION – KEY EXCHANGE



HYBRID ENCRYPTION – SESSION DATA

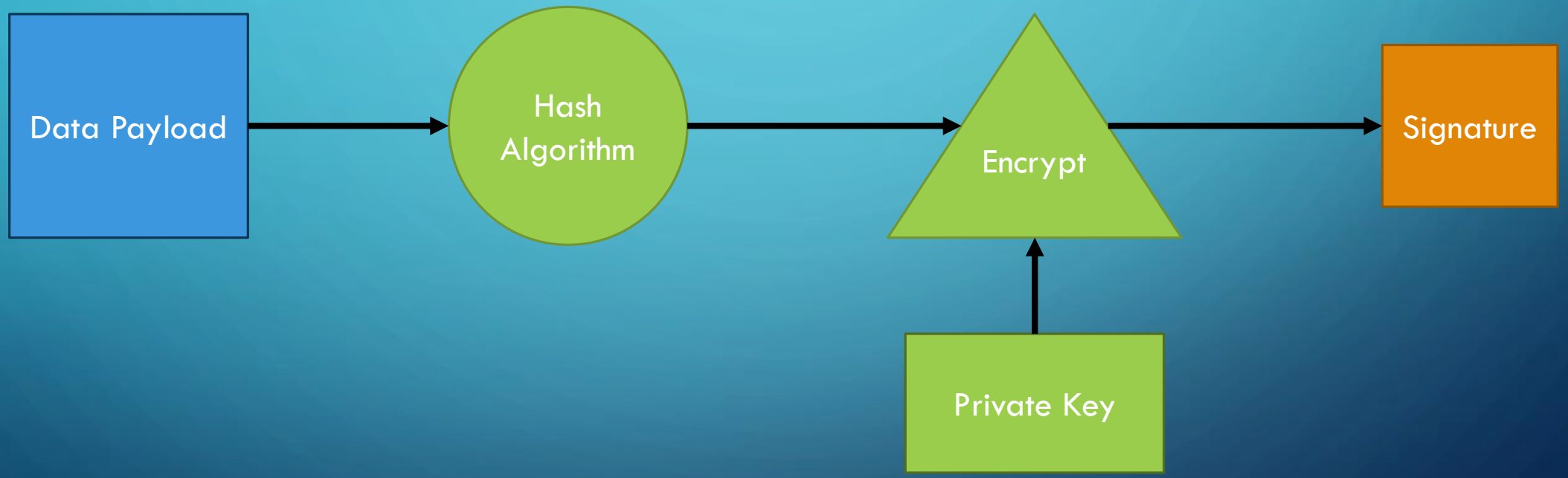


SIGNING

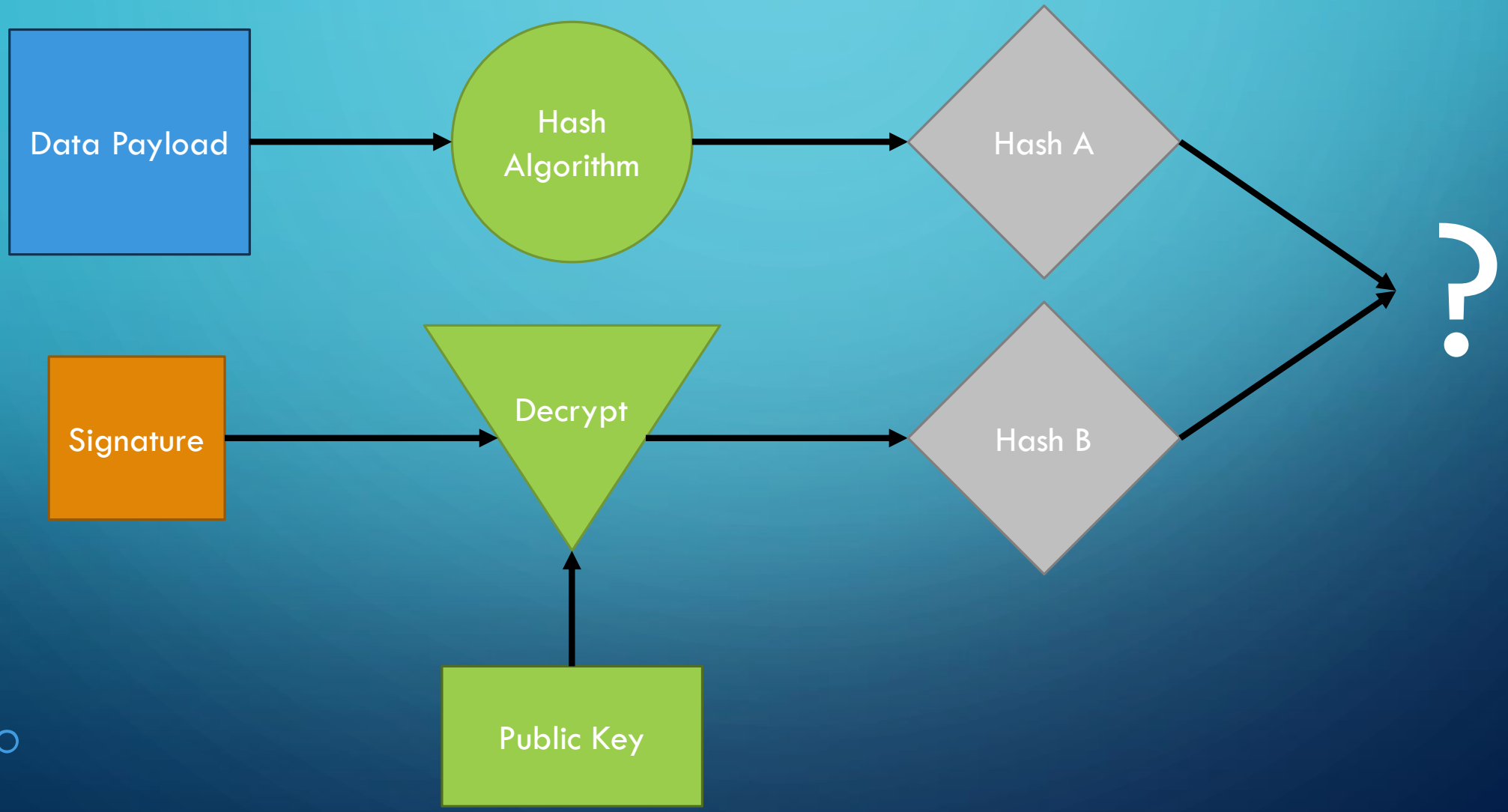
SIGNING

- Signing can provide:
  - Integrity checking
  - Proof of data source (trust)
- Blend of hashing and asymmetric encryption
- Commonly see signatures in:
  - Email
  - Certificate trust chains
  - JWT Access tokens (OAuth or OpenIdConnect)

SIGNING

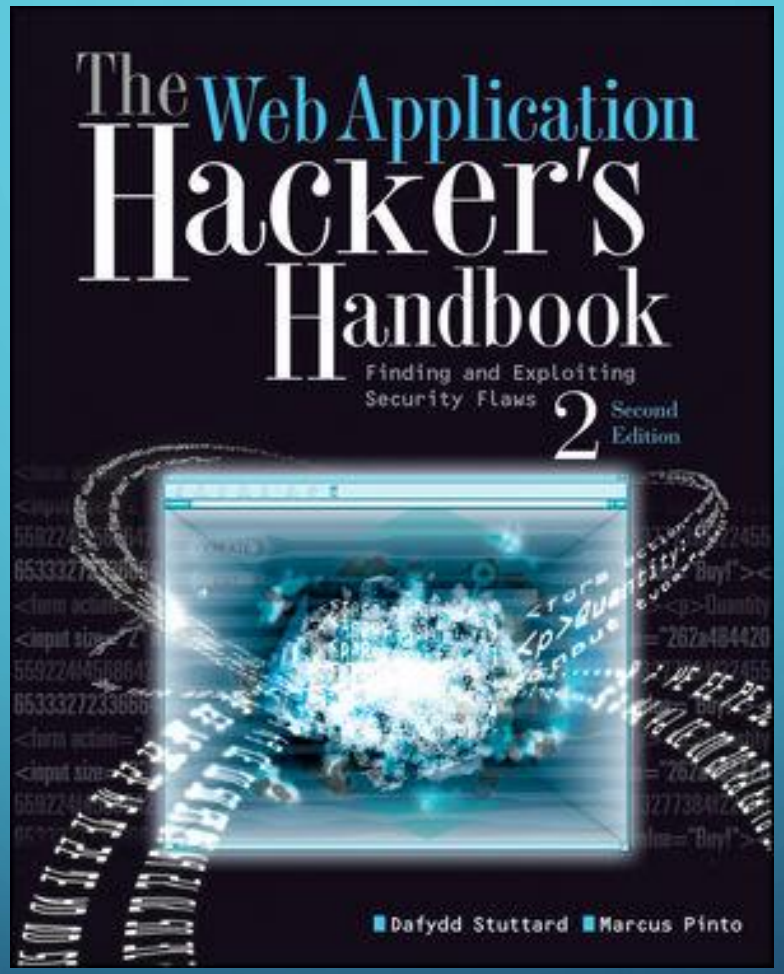

Data Payload
Hash Algorithm
Encrypt
Signature
Private Key

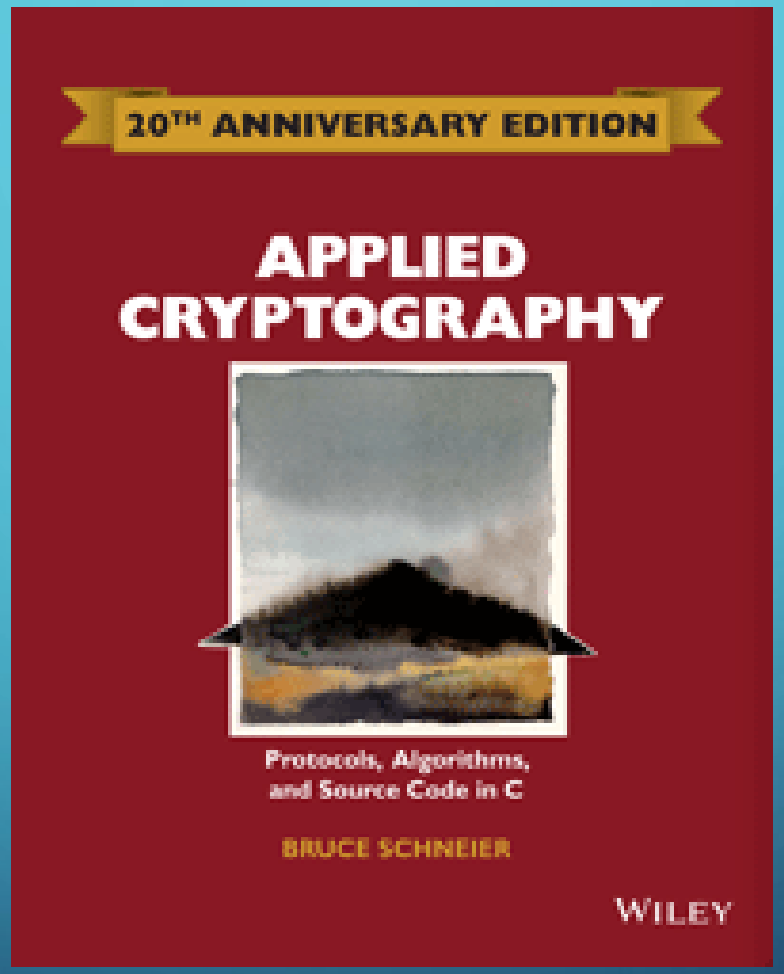
SIGNATURE VERIFICATION



GOALS OF CRYPTOGRAPHY

- Privacy – symmetric encryption
- Authentication – asymmetric encryption
- Integrity - hashing
- Nonrepudiation – hashing + asy


The Web Application Hacker's Handbook
Finding and Exploiting Security Flaws
2 Second Edition
Dafydd Stuttard
Marcus Pinto


20TH ANNIVERSARY EDITION
APPLIED CRYPTOGRAPHY
Protocols, Algorithms, and Source Code in C
BRUCE SCHNEIER
WILEY

QUESTIONS