Medii vizuale de programare
Curs 11

Conf. dr.ing. GENGE Béla

Universitatea "Petru Maior", Departamentul de Informatica
Tîrgu Mureș, Romania
bela.genge@ing.upm.ro

Servicii criptografice în .NET

- În .NET serviciile criptografice sunt disponibile în cadrul spațiului de nume `System.Security.Cryptography`.
- Ce ne oferă .NET:
  - Criptografie simetrică.
  - Criptografie asimetrică.
  - **Funcții hash și MAC.**
  - Generatoare de numere aleatoare.
  - Semnături digitale.

Managed vs unmanaged cryptography

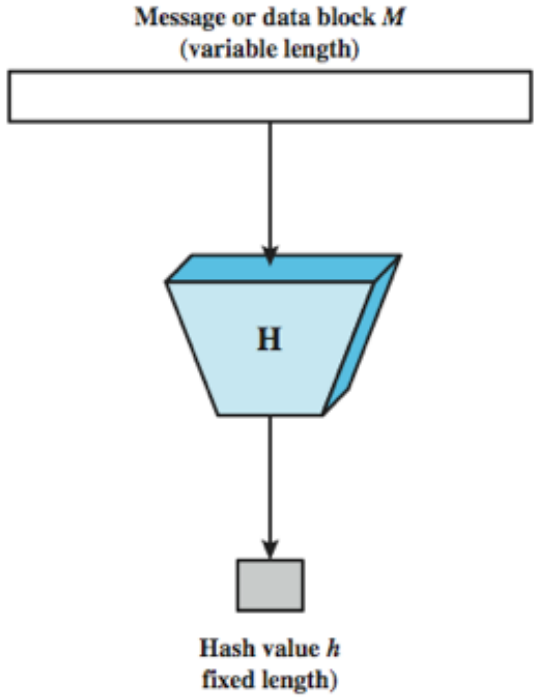
- Criptografia nativă (Win32/Win64) Windows este implementată în bibliotecile CryptoAPI - unmanaged code, din afara CLR. Main nou (după Windows Vista) în Cryptographic API Next Generation (CNG)
- .NET extinde CNG cu implementări managed pentru a face aplicațiile portabile.
- Dacă o aplicație .NET utilizează CNG, aceasta trebuie să fie disponibilă pe stațiile destinație.

.NET și FIPS 140-2

- .NET permite accesarea CNG, adică a criptografiei certificate FIPS 140-2 utilizând API-ul `unmanaged` (nativ).
  - Dezavantajul principal îl reprezintă dificultatea portării aplicațiilor pe platforme unde CNG nu e disponibil (e.g., Linux).
- .NET include și clase criptografice `managed`, dar acestea nu sunt certificate FIPS 140-2.
- API-ul `unmanaged` se distinge prin denumirea claselor ce include `CryptoServiceProvider` sau `CNG`.
- API-ul `managed` se distinge prin denumirea claselor ce includ `Managed`.

Funcții HASH

- Realizează o amprentă, en: digest, pentru o secvență de octeți.
- Aplicații variate:
  - Asigurarea integrității.
  - Generarea numerelor aleatoare.
  - Generarea cheilor criptografice.
- Exemplu calcul Hash pe imagini ISO (VirtualBox):
  `https://www.virtualbox.org/wiki/Downloads`



Funcții hash în .NET

- Clasa de bază: `System.Security.Cryptography.HashAlgorithm`.
- Ierarhia de clase: `https://msdn.microsoft.com/en-us/library/system.security.cryptography.hashalgorithm(v=vs.110).aspx`
- Clase care moștenesc `HashAlgorithm`:
  - `KeyedHashAlgorithm`.
  - `MD5`.
  - `RIPEMD160`.
  - `SHA1`.
  - `SHA256`.
  - `SHA384`.
  - `SHA512`.

Proprietăți și metode ale clasei `HashAlgorithm` (principale)

- Proprietăți (get/set pentru câmpurile clasei):
  - `Hash`.
  - `HashSize`.
  - `InputBlockSize`.
  - `OutputBlockSize`.

Câmpuri și metode ale clasei `HashAlgorithm` (principale)

- Metode principale:
  - `Clear()`: eliberarea tuturor resurselor.
  - `ComputeHash(Byte[])`: calculează hash-ul secvenței de octeți.
  - `ComputeHash(Byte[], Int32, Int32)`: calculează hash-ul secvenței de octeți pe o anume regiune a vectorului.
  - `ComputeHash(Stream)`: calculeaza hash-ul pe un stream.
  - `HashFinal()`: finalizează calcularea hash-ului.

Calcularea HASH-ului pe un fișier

- Se alege funcția HASH. Dacă nu există constrângeri cu privire la dimensiunea rezultatului, se recomandă funcțiile SHA-2 (SHA256, SHA384, SHA512).
- Se instanțiază (de exemplu): `SHA256CryptoServiceProvider`
- Se apelează `ComputeHash()` pe un obiect stream/byte vector.

Exemplu calculare HASH pe un fișier

Exemplu

```
FileStream fin = new FileStream("D:/source.pdf",
    FileMode.Open, FileAccess.Read);
HashAlgorithm cryptoProvider = new
    SHA256CryptoServiceProvider();
byte[] hashValue = null;
hashValue = cryptoProvider.ComputeHash(fin);
fin.Close();
MessageBox.Show("Hash: " +
    System.Text.Encoding.UTF8.GetString(hashValue));
MessageBox.Show("Length: " + hashValue.Length);
```

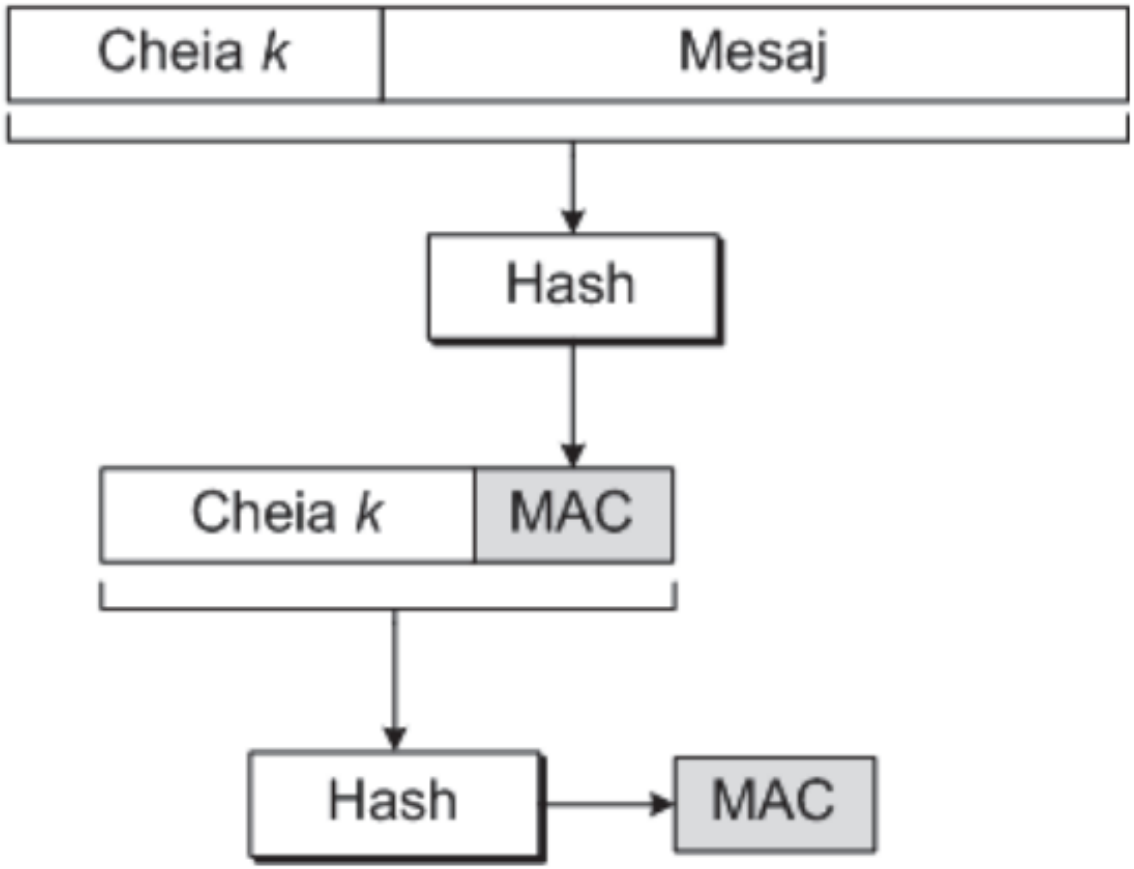
Exemplu calculare HASH pe un string

Exemplu

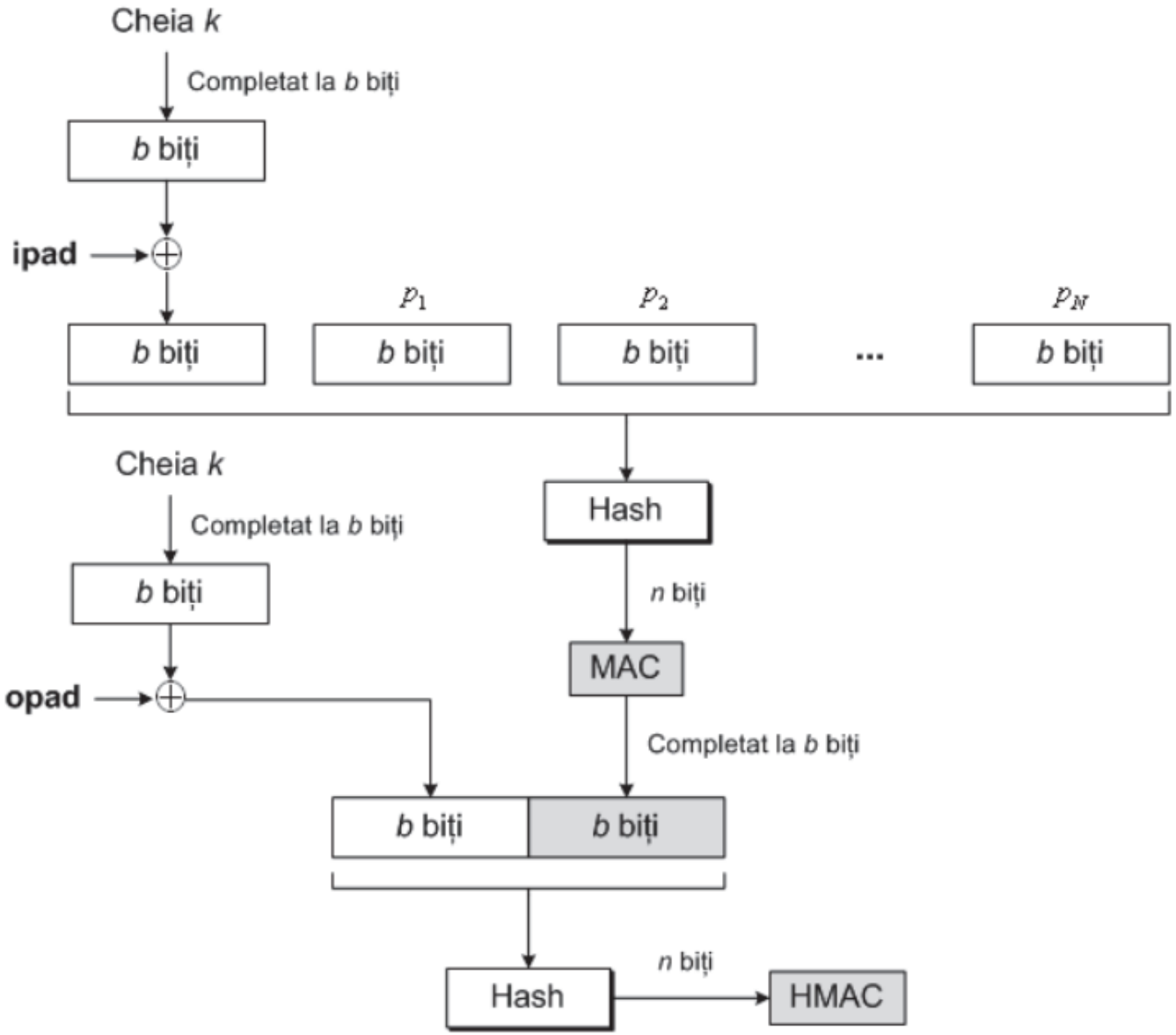
```
byte[] inp =
    System.Text.Encoding.UTF8.GetBytes(textBox1.Text);
HashAlgorithm cryptoProvider = new
    SHA256CryptoServiceProvider();
byte[] hashValue = null;
hashValue = cryptoProvider.ComputeHash(inp);
MessageBox.Show("Hash: " +
    System.Text.Encoding.UTF8.GetString(hashValue));
MessageBox.Show("Length: " + hashValue.Length);
```

MAC - Message Authentication Codes

- MAC = HASH(Cheie,HASH(Mesaj,Cheie)).
- Asigură: Integritatea + Autenticitatea datelor.



Standardul HMAC



MAC în .NET

- Clasa de bază: `System.Security.Cryptography.KeyedHashAlgorithm`.
- Ierarhia de clase: `https://msdn.microsoft.com/en-us/library/system.security.cryptography.keyedhashalgorithm(v=vs.110).aspx`
- Două implementări:
  - `System.Security.Cryptography.HMAC`.
  - `System.Security.Cryptography.MACTripleDES`.
- Implementări ale standardului HMAC:
  - `HMACMD5`.
  - `HMACRIPEMD160`.
  - `HMACSHA1`.
  - `HMACSHA256`.
  - `HMACSHA384`.
  - `HMACSHA512`.

Proprietăți și metode ale clasei `HMAC`

- Proprietăți (get/set pentru câmpurile clasei):
  - `BlockSizeValue`.
  - `Hash`.
  - `HashName`.
  - `HashSize`.
  - `InputBlockSize`.
  - `OutputBlockSize`.
  - `Key`.

Câmpuri și metode ale clasei `HMAC` (principale)

- Metode principale:
  - `Clear()`: eliberarea tuturor resurselor.
  - `ComputeHash(Byte[])`: calculează hash-ul secvenței de octeți.
  - `ComputeHash(Byte[], Int32, Int32)`: calculează hash-ul secvenței de octeți pe o anume regiune a vectorului.
  - `ComputeHash(Stream)`: calculeaza hash-ul pe un stream.
  - `HashFinal()`: finalizează calcularea hash-ului.

Calcularea HMAC pe un fișier

- Se alege funcția HASH. Dacă nu există constrângeri cu privire la dimensiunea rezultatului, se recomandă funcțiile SHA-2 (SHA256, SHA384, SHA512).
- Se generează o cheie criptografică de dimensiunea cel puțin 128 biți (de fapt se cere o secvență de octeți generați aleator, pe baza cărora funcția va genera automat cheia).
- Se instanțiază HMAC-ul utilizând unul din constructorii (exemplu pentru HMACSHA1):
  - `HMACSHA1()`: crează o nouă instanță cu o cheie generată automat.
  - `HMACSHA1(Byte[])`: crează o nouă instanță, cheia e transmisă ca parametru.
  - `HMACSHA1(Byte[], Boolean)`: crează o nouă instanță, cheia e transmisă ca parametru. Parametru `Boolean` controlează instanțierea:
    - true: `SHA1Managed`.
    - false: `SHA1CryptoServiceProvider`.

Exemplu calculare HMAC pe un fișier (cheia generată automat)

Exemplu

```
FileStream fin = new FileStream("D:/source.pdf",
    FileMode.Open, FileAccess.Read);
KeyedHashAlgorithm cryptoProvider = new HMACSHA256();
byte[] hmacValue = null;
hashValue = cryptoProvider.ComputeHash(fin);
fin.Close();
MessageBox.Show("HMAC: " +
    System.Text.Encoding.UTF8.GetString(hashValue));
MessageBox.Show("Length: " + hashValue.Length);
```

Exemplu calculare HMAC pe un fișier (cheia generată cu RNGCryptoServiceProvider)

Exemplu

```
byte[] cheiaSecreta = new byte[128];
RNGCryptoServiceProvider rng =
    new RNGCryptoServiceProvider();
rng.GetBytes(cheiaSecreta);
FileStream fin = new FileStream("D:/source.pdf",
    FileMode.Open, FileAccess.Read);
KeyedHashAlgorithm cryptoProvider = new
    HMACSHA256(cheiaSecreta);
byte[] hmacValue = null;
hmacValue = cryptoProvider.ComputeHash(fin);
fin.Close();
MessageBox.Show("HMAC: " +
    System.Text.Encoding.UTF8.GetString(hmacValue));
MessageBox.Show("Length: " + hmacValue.Length);
```

Criptografia asimetrică

- Asigură o serie de proprietăți: secretizare, integritate, autenticitate, non-repudiere.
- Aplicații: criptare/decriptare, semnătură digitală.
- Cifrul RSA.

Criptografia asimetrică în .NET

- Clasa de bază:
  `System.Security.Cryptography.AsymmetricAlgorithm`.
- Ierarhia de clase:
  `https://msdn.microsoft.com/en-us/library/system.security.cryptography.asymmetricalgorithm(v=vs.110).aspx`
- Clase care moștenesc `AsymmetricAlgorithm`:
  - `DSA`.
  - `ECDiffieHellman`.
  - `ECDsa`.
  - `RSA`.

Criptografia asimetrică în .NET

- Clasa : `System.Security.Cryptography.RSA`.
- Moștenită de:
  - `System.Security.Cryptography.RSACng`.
  - `System.Security.Cryptography.RSACryptoServiceProvider`.

Proprietăți și metode ale clasei RSA

- Proprietăți (get/set pentru câmpurile clasei):
  - `KeyExchangeAlgorithm`.
  - `KeySize`.
  - `LegalKeySizes`.
  - `SignatureAlgorithm`.

Câmpuri și metode ale clasei `RSA` (principale)

- Metode principale:
  - `Clear()`: eliberarea tuturor resurselor.
  - `Encrypt(Byte[], RSAEncryptionPadding)`: criptează o secvență de octeți.
  - `Decrypt(Byte[], RSAEncryptionPadding)`: decriptează o secvență de octeți.
  - `SignData(Byte[], HashAlgorithmName, RSASignaturePadding)`: semnează digital o secvență de octeți.
  - `VerifyData(Byte[], Byte[], HashAlgorithmName, RSASignaturePadding)`: verifică semnătura digitală.