VIRIDIAN
Powering the Future

Encryption, Certificates and SSL

DAVID COCHRANE

PRESENTATION TO BELFAST OWASP CHAPTER

OCTOBER 2018

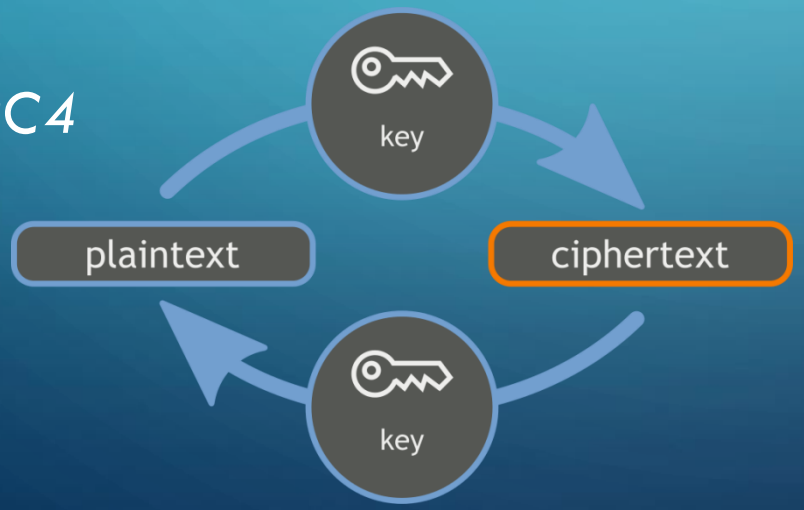
Agenda

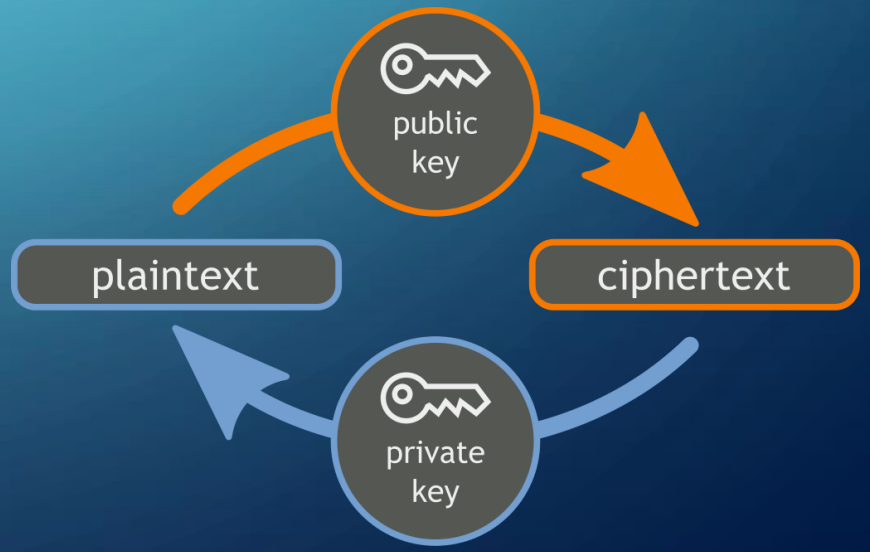
- Basic Theory: encryption and hashing
- Digital Certificates
- Tools for Digital Certificates
- Design Patterns
- Case Study – Build your own CA – Slides not published

Encryption Basics

- AES
- Blowfish
- Twofish
- *3DES*
- *RC2, RC4*

- RSA
- Elliptic Curve





Cryptographic Hashing

- Converts a large amount of data to a “representative” number
  - *MD5 – 128 bits*
  - *SHA1 – 160 bits*
  - SHA2 / SHA256 – 256 bits
  - SHA384, SHA512 – 384, 512 bits respectively
- One-way process
- Used to verify that two files or strings are the same – without checking both byte by byte
- Safe storage of passwords
- Importance of “Salt”
- Digital signature = EncryptUsingPrivateKey (Hash(Data))

What's a Digital Certificate?

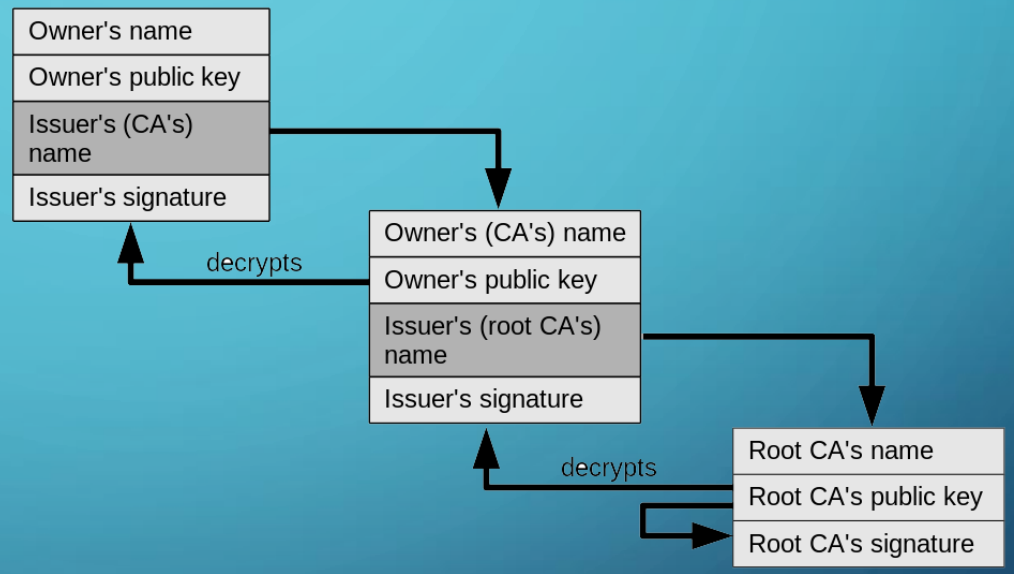
- Data that represents an entity or object and can be used to verify its identity
- Attributes are defined by X.509

Issued to: CN=David Cochrane,O=Viridian Group,C=GB,L=Belfast
Issued by: CN=Certifying Authority,O=Your favourite CA,…

- Subject and issuer (X.500 format)
- Subject public key
- Start and end dates
- Serial number and hash
- Usage (Basic and enhanced)
- Alternate names (DNS or IP)
- Subject key ID and authority key ID
- Revocation (CRL and/or OCSP)
- Digital signature of issuer
- …

What Makes a Certificate Trusted – PKI

- Certificate Chain
  - Certifying Authority (CA)
  - Intermediate Certificates
- Start Date and Expiry
- Revocation
  - CRL or OCSP



Windows and Digital Certificates

- Windows Certificates Stores
  - Two physical stores: User and Machine
  - Logical stores in each: Personal, Trusted Root Certification Authorities, Other People, Trusted People, Trusted Publishers, Intermediate Certification Authorities, Active Directory User Object, …

Store type Active Directry, location LDAP:///'dn'?userCertificate

- Windows Keystores
  - Private key is stored in a keystore separately from the certificate
  - Separate keystore for each Crypto service provider, e.g.
    - Microsoft Enhanced RSA and AES Cryptographic Provider
    - Microsoft Enhanced DSS and Diffie-Hellman Cryptographic Provider
  - User keystores protected by user key, which is derived from user's password
  - Machine keystores protected by machine key – needs local admin to access
  - Private key memory is protected by Windows Crypto system, smart card or TPM chip

Digital Certificate Formats

- DER – binary encoded using ASN.1 (Windows file type .cer)
- PEM – Base64 encoded, separate KEY file contains encrypted key or encrypted within the PEM file
- File type .crt can be either a DER or PEM file
- PKCS12 store (PFX or P12) – contains private key protected by password, can contain multiple certificates, e.g., complete certificate chain
- PKCS7 store – contains multiple certificates similar to DER (file type .p7b)
- CSR – certificate signing request, no key details or digital signature
- JKS – Java keystore for users of Sun's Java crypto library

Certificate Signing Requests

- Details of the certificate to be signed, similar to CER format
- Private key is stored on the server that generated the request
- Usually uploaded to Certifying Authority's web site so that signed certificate can be downloaded
- Certificate signing requests can be generated by IIS, Windows certificate manager or OpenSSL
  - Key length, usage, algorithm, alternate names, …
- Private key is matched with the signed certificate when it is installed on the server

Certificate Attributes - Windows

Certificate

General | Details | Certification Path

Certificate Information

This certificate is intended for the following purpose(s):

- Proves your identity to a remote computer
- Protects e-mail messages
- Ensures software came from software publisher
- Protects software from alteration after publication

Issued to: David Cochrane

Issued by: Viridian Group Certifying Authority

Valid from 10-Oct-16 **to** 10-Oct-19

Install Certificate... | Issuer Statement

OK

Certificate

General | Details | Certification Path

Show: <All>

Field	Value
Serial number	06b8fad579613fe2d0bffdb95...
Signature algorithm	sha512RSA
Signature hash algorithm	sha512
Issuer	Viridian Group Certifying Auth...
Valid from	Monday, October 10, 2016 4:...
Valid to	Thursday, October 10, 2019 4...
Subject	David.Cochrane@viridiangrou...
Public key	RSA (2048 Bits)

E = David.Cochrane@viridiangroup.co.uk
CN = David Cochrane

Field	Value
Subject	David.Cochrane@viridiangrou...
Public key	RSA (2048 Bits)
Public key parameters	05 00
Authority Key Identifier	KeyID=73ff9a3c7f004a61598...
Subject Key Identifier	c4c74ff1a750d4bfa66cd6b59b...
Enhanced Key Usage	Client Authentication (1.3.6.1....
Thumbprint	904e74cfcfe368a6f1794ed97f...

Client Authentication (1.3.6.1.5.5.7.3.2)
Secure Email (1.3.6.1.5.5.7.3.4)
Code Signing (1.3.6.1.5.5.7.3.3)

Certificate

General | Details | Certification Path

Certification path

Viridian Group Certifying Authority
David Cochrane

View Certificate

Certificate status:

This certificate is OK.

OK

Certificate Attributes – OpenSSL

```
Data:
    Version: 3 (0x2)
    Serial Number:
        06:b8:fa:d5:79:61:3f:e2:d0:bf:fd:b9:56:6d:04:98
Signature Algorithm: sha512WithRSAEncryption
    Issuer: OU = Technology and Change Team, O = Viridian Group Limited, CN = Viridian Group Certifying Authority
    Validity
        Not Before: Oct 10 15:19:12 2016 GMT
        Not After : Oct 10 15:19:12 2019 GMT
    Subject: CN = David Cochrane, emailAddress = David.Cochrane@viridiangroup.co.uk
    Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
            Public-Key: (2048 bit)
    X509v3 extensions:
        X509v3 Extended Key Usage: critical
            TLS Web Client Authentication, E-mail Protection, Code Signing
        X509v3 Subject Key Identifier:
            C4:C7:4F:F1:A7:50:D4:BF:A6:6C:D6:B5:9B:A1:6A:37:F1:C1:EA:21
Signature Algorithm: sha256WithRSAEncryption
```

Certificates and SSL/TLS

- Server certificate provided during negotiation must be trusted by the client browser. Firefox, Safari user their own certificate stores
- Wildcard and Subject Alternate Name certificates allow one certificate for multiple sites
- Certificates for an Internet site can be requested from public CA via a CSR
  - Can't use internal server name or IP address
- Domain Validated certificates vs Extended Validation certificates
- One year up to five years validity
- https://letsencrypt.org – free SSL certificates, uses HTTP validation
- https://ssllabs.com/ssltest

Certificates and SSL/TLS

- Make sure you specify enough X.500 attributes: CN, C, O, (OU), (L)
- Allow for all of the possible names as Subject Alternate Names or use a wildcard certificate
- Specify RSA 2048 or better for encryption and SHA256 or better for hashing
- Alternatively, ECC 256 bit is acceptable
- Specify Microsoft Enhanced Crypto Provider to store the private key
- For IIS install the certificate into the Machine Personal store (or install using IIS Server Certificates option)
- Update config file for web servers other than IIS: Apache, Tomcat, Weblogic, …

Sample Certificate - Wikipedia

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            08:30:94:62:d1:fe:a6:0a:e0:ba:bf:f5:ef:8b:c5:45
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = US, O = DigiCert Inc, OU = www.digicert.com, CN = DigiCert SHA2 High Assurance Server CA
        Validity
            Not Before: Dec 21 00:00:00 2017 GMT
            Not After : Jan 24 12:00:00 2019 GMT
        Subject: C = US, ST = California, L = San Francisco, O = "Wikimedia Foundation, Inc.", CN = *.wikipedia.org
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Authority Key Identifier:
                keyid:51:68:FF:90:AF:02:07:75:3C:CC:D9:65:64:62:A2:12:B8:59:72:3B
            X509v3 Subject Key Identifier:
                6E:AD:11:B1:EE:67:1C:EB:54:DD:F2:2A:66:54:C1:BE:D0:3B:28:39
            X509v3 Subject Alternative Name:
                DNS:*.wikipedia.org, DNS:wikipedia.org, DNS:*.m.wikipedia.org, DNS:*.zero.wikipedia.org, DNS:wikimedia.org, DNS:*.wikimedia.org, DNS:*.m.wikimedia.org, DNS:*.planet.wikimedia.org, DNS:mediawiki.org,
DNS:*.mediawiki.org, DNS:*.m.mediawiki.org, DNS:wikibooks.org, DNS:*.wikibooks.org, DNS:*.m.wikibooks.org, DNS:wikidata.org, DNS:*.wikidata.org, DNS:*.m.wikidata.org, DNS:wikinews.org, DNS:*.wikinews.org, DNS:*.m.wikinews.org,
DNS:wikiquote.org, DNS:*.wikiquote.org, DNS:*.m.wikiquote.org, DNS:wikisource.org, DNS:*.wikisource.org, DNS:*.m.wikisource.org, DNS:wikiversity.org, DNS:*.wikiversity.org, DNS:*.m.wikiversity.org, DNS:wikivoyage.org,
DNS:*.wikivoyage.org, DNS:*.m.wikivoyage.org, DNS:wiktionary.org, DNS:*.wiktionary.org, DNS:*.m.wiktionary.org, DNS:wikimediafoundation.org, DNS:*.wikimediafoundation.org, DNS:*.m.wikimediafoundation.org, DNS:wmfusercontent.org,
DNS:*.wmfusercontent.org, DNS:w.wiki
            X509v3 Key Usage: critical
                Digital Signature
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
```

Sample Certificate - Wikipedia

```
X509v3 CRL Distribution Points:
            Full Name:
              URI:http://crl3.digicert.com/sha2-ha-server-g6.crl
            Full Name:
              URI:http://crl4.digicert.com/sha2-ha-server-g6.crl
X509v3 Certificate Policies:
            Policy: 2.16.840.1.114412.1.1
              CPS: https://www.digicert.com/CPS
            Policy: 2.23.140.1.2.2

          Authority Information Access:
            OCSP - URI:http://ocsp.digicert.com
            CA Issuers - URI:http://cacerts.digicert.com/DigiCertSHA2HighAssuranceServerCA.crt

          X509v3 Basic Constraints: critical
            CA:FALSE
          CT Precertificate SCTs:
            Signed Certificate Timestamp:
              Version   : v1 (0x0)
              Log ID    : BB:D9:DF:BC:1F:8A:71:B5:93:94:23:97:AA:92:7B:47:
                          38:57:95:0A:AB:52:E8:1A:90:96:64:36:8E:1E:D1:85
              Timestamp : Dec 21 18:11:19.631 2017 GMT
              Extensions: none
              Signature : ecdsa-with-SHA256
Signed Certificate Timestamp:
              Version   : v1 (0x0)
              Log ID    : 87:75:BF:E7:59:7C:F8:8C:43:99:5F:BD:F3:6E:FF:56:
                          8D:47:56:36:FF:4A:B5:60:C1:B4:EA:FF:5E:A0:83:0F
              Timestamp : Dec 21 18:11:19.720 2017 GMT
              Extensions: none
              Signature : ecdsa-with-SHA256
Signature Algorithm: sha256WithRSAEncryption
```

Important Tools for Digital Certificates

- Windows
  - Certificate Manager (MMC or certmgr.msc) – GUI for managing certificates in Windows
  - Certificate Utility – Windows certificate services
  - MAKECERT – Basic tool to create certificates, part of Windows SDK
  - SignTool – Code signing, part of Windows SDK
  - Encrypted File System – the easy way to encrypt files
  - ASPNET_REGIIS – encrypts / decrypts .Net web config files
- Cross-platform
  - OpenSSL – powerful command line tool to do almost anything with certificates
  - Keytool – creates and modifies JKS files (part of Java Developer Kit)

OpenSSL Commands

X509 – display and convert DER and PEM certificates

PKCS12 – create, verify and display PFX files

REQ – create and display certificate requests

OCSP – check certificate validity using OCSP

GENRSA – create an RSA key

ENC – encrypt or decrypt

CA – functions to act as a basic certifying authority

S_CLIENT – make an SSL / TLS connection to a web site, FTPS server or SMTP server

Libraries

- Windows CryptoAPI (some functions now deprecated W10 / WS2016)
- Windows CNG (Cryptography API Next Generation) – ECC support
- .Net System.Security.Cryptography namespace
- OpenSSL
- Java Cryptography Architecture
- Bouncy Castle
- …

Design Patterns

- .Net web site – storing application passwords securely
- Windows application – storing passwords or SSH keys securely
- Encrypt a file using a certificate
- Using a certificate for web site authentication
- Validate user login and password
- Verify user identity in a client application and an Intranet site

.Net web site – storing passwords securely

- Method 1 (recommended)
  - Store password in web config file and use ASPNET_REGIIS as follows:

    ASPNET_REGIIS -pe “PasswordSection” Webroot (to encrypt)

    ASPNET_REGIIS -pd “PasswordSection” Webroot (to decrypt)
- Method 2
  - Store the password in a separate file that has been manually encrypted using certificate
  - Install the certificate in the machine certificate store
  - In your application code load the certificate, and use private key to decrypt the contents of the file

Storing passwords or SSH keys securely

- Method 1 (recommended for server applications)
  - Use Windows EFS to encrypt the file using the credentials of the account the application will run under
- Methods for client applications
  - Use a secure web service to retrieve the password or SSH key
    - Configure the web service to run under the user's credentials then retrieve those
  - OR Use an encrypted Kerberos connection to a server-based application (see later) to retrieve password or key

Encrypt a File using a Certificate – Method 1

- Access certificate from store
- Obtain public key
- Read file contents
- Encrypt using public key
- Save file contents

```
X509Store store = new X509Store ("My");

X509Certificate2Collection collection = store.Certificates;

X509Certificate2 certificate =
   collection.Find(FindBySubjectDistinguishedName, "Encryptor")[0];

RSACryptoServiceProvider encryptor = certificate.PublicKey;

//  Read file contents into byte[] clearData

encryptor.Encrypt(clearData, encryptedData);

//  Overwrite file with encrypted data
```

Encrypt a File using a Certificate – Method 2

- Create a random symmetric key
- Access certificate from store
- Obtain public key
- Read file contents
- Encrypt symmetric key using public key
- Encrypt file using symmetric key
- Save encrypted symmetric key and encrypted file contents

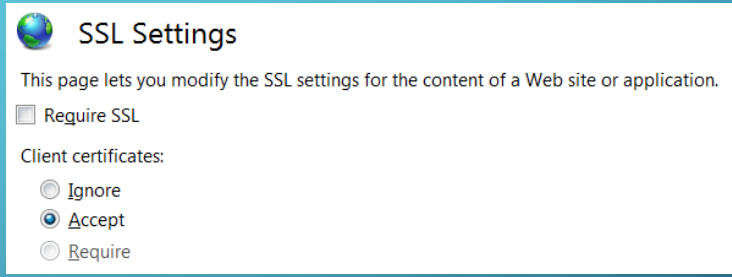
```
AESCryptoServiceProvider aes = new AESCryptoServiceProvider();

aes.GenerateKey();

X509Store store = new X509Store ("My");

X509Certificate2Collection = store.Certificates;

X509Certificate2 certificate =
    collection.Find(FindBySubjectDistinguishedName, "Encryptor")[0]

RSACryptoServiceProvider encryptor = certificate.PublicKey;

//  Read file contents

encryptor.Encrypt(aes.Key, encryptedKey);

ICryptoTransform aesEncrypt = aes.CreateEncryptor(aes.Key, aes.IV);

//  Encrypt data a block at a time using aesEncrypt

//  Overwrite file with encrypted key and data
```

Using a Certificate for Authentication

CLIENT

```
WebRequestHandler handler = new WebRequestHandler();

X509Certificate certificate = GetClientCertFromStore();

handler.ClientCertificates.Add(certificate);

HttpClient client = new HttpClient(handler);

HttpResponse response = await client.GetAsync(URL);
```

SERVER



Validate User Login and Password – Method 1

- Retrieve user's hashed password
- Hash supplied password
- Compare them

```
bool function IsPasswordValid (string username, string enteredPassword)
{
//  SELECT UserHash FROM Users WHERE Login = ?username
SHA256CryptoServiceProvider hasher = new SHA256CryptoServiceProvider();
hasher(Encoding.UFT8.GetBytes(enteredPassword + salt), enteredHash);
for (int i=0; i < userHash.ArraySize; i++)
        if (enteredHash[i] != userHash[i]) return false;
return true;
}
```

Validate User Login and Password – Method 2

- Retrieve user's encrypted password
- Decrypt it
- Compare with entered password

```
bool function IsPasswordValid(string username, string enteredPassword)
{
//  SELECT encryptedPassword FROM Users WHERE Login = ?username
key.Decrypt(encryptedPassword, clearPasswordBytes);
string clearPassword  = Encoding.UTF8.GetString(clearPasswordBytes);
return (clearPassword == enteredPassword);
}
```

- **What are the two coding flaws?**

Verify User Identity

Client Application

```
WindowsIdentify wi = WindowsIdentity.GetCurrent();

WindowsPrincipal wp = new WindowsPrincipal(wi);

string username = wp.Identity.Name;
```

Intranet Web Application

```
//  Ensure IIS is configured to use Windows authentication and the
web application pool runs under the user's identity

string username = Page.User.Identity.Name
```

Questions

VIRIDIAN
Powering the Future

?