kamstrup

Entschlüsseln einer KEM- oder KEM2-Datei

kamstrup

1 Inhalt

Der Beispielcode in diesem Dokument basiert auf C# und .NetCore.

1 Die KEM2-Datei

Die KEM2-Datei ist, wie eine KEM-Datei, ein Dateiformat für alle in diesem Dokument beschriebenen Zwecke, aber mit der Eigenschaft, von Kamstrup mit seinem privaten Schlüssel signiert zu werden.

Durch die Verifizierung dieser digitalen Signatur werden die Authentizität dieses Dokument, das von Kamstrup erhalten wird, und die Integrität der Datei selbst sichergestellt werden. Um die Signatur zu verifizieren, müssen Sie das Public-Key-Zertifikat im persönlichen Ordner vom Local Machine Certificate Manager speichern. Das Zertifikat können Sie hier herunterladen:

https://eks.kamstrup.com/certificates/Eks.Signing.PublicKey.cer

Um einen gültigen Zertifizierungspfad zu haben, müssen Sie auch das Zertifikat im Trusted Root Certification Authorities Store installieren:

https://eks.kamstrup.com/certificates/Eks.Signing.HCWSSRootCA.cer

https://eks.kamstrup.com/certificates/Eks.Signing.CA.crt

Und das folgende Zertifikat muss im Ordner Intermediate Certification Authorities gespeichert werden:

https://eks.kamstrup.com/certificates/Eks.Signing.C1.crt

Sehen Sie den Abschnitt “Die Signatur validieren” für Informationen darüber, wie Sie das Public-Key-Zertifikat für die Validierung der Signatur verwenden.

2 KEM-Datei herunterladen

Laden Sie die Datei vom Encryption Key Service-Portal herunter. Wählen Sie KEM, und geben Sie ein Passwort ein, um die Datei zu sichern.

Für weitere Informationen über das Herunterladen einer KEM-Datei, siehe die Superuseranleitung.

3 Daten extrahieren

Die heruntergeladene Datei sollte wie das nachstehende Beispiel aussehen und die Erweiterung „**.zip.kem**“ oder „**.zip.kem2**“ haben.

```
20210407_1119DownloadMeters.zip.kem
20210812_1131_selected_DownloadDevices.zip.kem2
```

Die Erweiterung löschen

Benennen Sie die Datei um, und löschen Sie die Erweiterung „**.kem**“ oder „**.zip.kem2**“, um sie in eine ZIP-Datei zu konvertieren.

Entpacken

Der Ausgabeordner wird wie die nachstehenden Dateien aussehen:

- 363A6D7848DF4882B0991674191A9B84.kem
- meter_information_file.xsd

Schemadatei

Die Datei mit der Erweiterung „**.xsd**“ enthält das Schema der Daten, wenn sie entschlüsselt worden sind. Mit dieser Datei können Sie validieren, ob die Ausgabe das erwartete Format hat.

Beachten Sie, dass das Schema für einige Einheiten verschieden sein kann (device_information_file.xsd).

Datendatei

Die Datei mit der Erweiterung „**.kem**“ enthält die Daten. Sie ist eine XML-Datei, die wie folgt aussehen soll:

```
<?xml version=”1.0” encoding=”utf-8”?><EncryptedData
Type=”http://www.w3.org/2001/04/xmlenc#aes128-cbc”
xmlns=”http://www.w3.org/2001/04/xmlenc#”>
<EncryptionMethod Algorithm=”http://www.w3.org/2001/04/xmlenc#aes128-cbc” />
     <CipherData>
          <CipherValue>
              Nd7AyPuTPjsRgvS0l3ExFLqmsAgKDzOQYBQToY23/4g7ArcUF3RXEuYhwY2Uskdlq
              knRyvmZSuTlrNqctiafLB5c9pgpEZ0KydL1Sm26OmhsZjMzg2qhcLPoA3DV/aA97s
              aotIlxYk3Pq2DmpkJtA8Mw4Kh57TirNOcFB3mhTN4V3GjwevmNib2nZ22EvU0V17i
              GHCQPNMl3VN5lK5Ev8QnNT4Mjtlr3d0P7iWkO3w7H35lYPdoDZzaZiCluCTxKUZMW
              TGTRRcUHgvDifSmCXhmpZS7f4AA+3S9x2LUB6vdHlQYh5FxQT/tsy/mLVphzDYXEE
              Ej+TLG7Y3g59T0WVf6h+x8J0Rh5SGWfsV58F6+276cDkfMX7A8KanU6owlW6MFg/s
              5s1RyK397mjDw9UJh9bIJuTc5Yeq6D8I3K5c5EyF+MYDGUo+SxwXbrX/j0UJkmua/
              w4yhzBSvccF0yGQXS4QpVEK8yq+dRHvtpac8ZCXRS2OLP+eZt8qCQJDmpJ/M02kgx
              UAo9Ocd30EBokXLxuwjOqxeZ/ZNTG3vFtn89dMhW+QtL8k8UL1hBkvaCClApLoi38
              RVCFd/D9LIGJBDtsa20eu9miHX5CraLaLzsoXYXE8ijSU5u2we4N3mSGBzJWam/2i
              JkHIxrGpRYqovNDHT7BHd/EVwGlzbekbVueWm8fjcnkXGY5GtxnciuIUJ4rvfvdmq
              tcFXOR5iAx0RK25Y0/I6mmj3+S0f24hGatmzNfVWmNqFXmybO717i+sQMfyflTli/
              RY6Eeb/0G3qABv+NIWJP+8p2CSDCC59LmrsDWFtDxsEBn+RQnTMT+vh4VDPOFEuM1
              SNlvvHozlwsIV0jdQxw1muc7yVzeaHZoGIJe+fculeB4VLsaGxZVOHnd3R6iwYc1v
              z7r1KiDXlh/F0zORtaAowoqglPNfQ=
          </CipherValue>
     </CipherData>
</EncryptedData>
```

Datei in Daten konvertieren

Die verschlüsselten Daten stellen den Wert des Inhalts des Tags „**CipherValue**“ in der Datendatei dar. Laden Sie die Datei als XML, suchen Sie nach dem Tag, und konvertieren Sie sie in Byte-Array.

```
var encryptedXmlDocuent = new XmlDocument();
encryptedXmlDocuent.Load(kemFilePath);
var encryptedElement = encryptedXmlDocuent.GetElementsByTagName(“CipherValue”)[0]
as
XmlElement;
var encryptedData = Convert.FromBase64String(encryptedElement.InnerText);
```

4 Daten entschlüsseln

Sicheres Passwort

Sie müssen das Passwort, das beim Herunterladen der Datei eingegeben wurde, in den Typ SecureString konvertieren.

```
SecureString pwd = plainPwd.ConvertToSecureString();
public static SecureString ConvertToSecureString(this string unsecurePassword)
{
    if (unsecurePassword == null)
    {
        throw new ArgumentNullException(“unsecurePassword”);
    }

    var securePassword = new SecureString();
    foreach (var c in unsecurePassword)
        securePassword.AppendChar(c);
    securePassword.MakeReadOnly();
    return securePassword;
}
```

Entschlüsseln

Mit dem Schlüssel und den verschlüsselten Daten kann der Entschlüssler die Daten entschlüsseln und konvertieren.

Dies ist als unsicher gekennzeichnet. Deshalb muss das Projekt selbst als zugelassen gekennzeichnet werden, bevor ein unsicherer Code verwendet werden kann.

```
public static unsafe byte[] GetDecryptedData(SecureString key, byte[] textToD-
ecrypt)
{
    const int keyLength = 0x10;
    byte[] decryptedText;
    var keyAs16Bytes = new byte[keyLength];

    //validations

    var unmanagedBytes = Marshal.SecureStringToGlobalAllocAnsi(key);
    try
    {
        byte* byteArray = (byte*)unmanagedBytes;
        var pEnd = byteArray;
        while (*pEnd++ != 0){}
        var length = (int)((pEnd - byteArray) - 1);
        for (var i = 0; i < length; ++i)
        {
            keyAs16Bytes[i] = *(byteArray + i);
        }

        using (var alg = new RijndaelManaged())
        {
            //User cipher block chaning
            alg.Mode = CipherMode.CBC;
            alg.Padding = PaddingMode.PKCS7;

            // 128 bit key
            alg.KeySize = 0x80;
            alg.BlockSize = 0x80;

            // Secret key
            alg.Key = keyAs16Bytes;
            // Initialation Vector
            alg.IV = keyAs16Bytes;

            decryptedText = alg.CreateDecryptor().TransformFinalBlock(textToD-
ecrypt, 0, textToDecrypt.Length);
        }
    }
    catch (CryptographicException e)
    {
        throw new Exception(“CryptographicException while decrypting document.”,
          e);
    }
    finally
    {
        Marshal.ZeroFreeGlobalAllocAnsi(unmanagedBytes);
        Array.Clear(keyAs16Bytes, 0, keyLength);
    }

    return decryptedText;
}
```

Validierungen

Im Folgenden werden einige der empfohlenen Überprüfungen angezeigt, die vor der Entschlüsselung gemacht werden können, um weitere Ausnahmen zu vermeiden.

```
   if (textToDecrypt == null || textToDecrypt.Length == 0)
    {
        throw new Exception("No encrypted data");
    }

    if (key == null || key.Length == 0)
    {
        throw new Exception("Key cannot be null or empty");
    }

    if (key.Length > keyLength)
    {
       throw new Exception("Key size is wrong. The key can not be larger than " + key-
Length);
 }
```

Ausgabe

Durch das Konvertieren der Ausgabe in einen String, wird es einfacher, sie zu lesen, und ein XML-Dokument kann erstellt werden.

```
decrtyptedText = Encoding.UTF8.GetString(decryptedData);
```

```
<MetersInOrder orderid="" schemaVersion="2.0">
    <Meter>
        <MeterNo>78127978</MeterNo>
        <SerialNo>78127978</SerialNo>
        <EncKeys>
            <DEK>00000000000000000000000000000004</DEK>
        </EncKeys>
        <MeterName>MC602</MeterName>
        <ConsumptionType>Cooling</ConsumptionType>
        <ConfigNo>510002424003</ConfigNo>
        <ProgramNo>44458458</ProgramNo>
        <TypeNo>602A00000075DA</TypeNo>
        <VendorId>KAM</VendorId>
    </Meter>
    <Meter>
        <MeterNo>78177775</MeterNo>
        <SerialNo>78177775</SerialNo>
        <EncKeys>
            <DEK>00000000000000000000000000000004</DEK>
        </EncKeys>
        <MeterName>MC602</MeterName>
        <ConsumptionType>Heat</ConsumptionType>
        <ConfigNo>217002424003</ConfigNo>
        <ProgramNo>34451451</ProgramNo>
        <TypeNo>602C03870A1212</TypeNo>
        <VendorId>KAM</VendorId>
    </Meter>
</MetersInOrder>
```

5 Die Signatur validieren

Das Zertifikat abrufen

Erst müssen Sie den Maschinenzertifikatspeicher durchlaufen, um nach dem Public-Key-Zertifikat zu suchen. Dafür müssen Sie einfach den Daumenabdruck des Zertifikats kennen, um es zu erkennen (in diesem Fall "14d7305cadf3c982a390c-94cc6e0054091b63531") und dann den X509Store loopen, um es abzurufen und es als eine X509Certificate-Klasse zu speichern. Beide Klassen stammen aus dem Namensraum System.Security.Cryptography.X509Certificates.

```
var certThumbprint = "14d7305cadf3c982a390c94cc6e0054091b63531";

    X509Certificate2 cert = null;
    using (var store = new X509Store(StoreLocation.LocalMachine))
            {
                store.Open(OpenFlags.ReadOnly);

                foreach (var certificate in store.Certificates)
                {
                    if (certificate.Thumbprint == certThumbprint.ToUpper())
                    {
                        cert = certificate;
                        break;
                    }
                }
                store.Close();
            }
```

Das Zertifikat für die Validierung der Signatur verwenden

Beim Entschlüsseln einer KEM2-Datei ist der entschlüsselte Text verschieden sein:

Sie müssen die Klasse SignedXml vom Namensraum SignatureSystem.Security.Cryptography.Xml entnehmen, das “Signature”-Tag in dieser Klasse laden, und die CheckSignature-Methode mit dem öffentlichen Schlüssel aus der X509Certificate-Klasse abrufen.

```
var decryptedXmlDocument = new XmlDocument();
decryptedXmlDocument.LoadXml(decryptedText);

var signedXml = new SignedXml(decryptedXmlDocument);
signedXml.LoadXml((XmlElement)decryptedXmlDocument.GetElementsByTagName(“Signature”)[0]);
var verifySignature = signedXml.CheckSignature((RSA)cert.PublicKey.Key);
```

Kamstrup • 55123207_B1_DE_10.2021

Kamstrup A/S, Deutschland
Werderstraße 23-25
D-68165 Mannheim
T: +49 621 321 689 60
F: +49 621 321 689 61
info@kamstrup.de
kamstrup.com

Kamstrup Austria GmbH
Handelskai 94 – 96
Millennium Tower – 32. OG, TOP 321
A-1200 Wien
T: +43 1 9073 666
info-at@kamstrup.com
kamstrup.com

Kamstrup A/S, Schweiz
Industriestrasse 47
CH-8152 Glattbrugg
T: +41 43 455 70 50
F: +41 43 455 70 51
info@kamstrup.ch
kamstrup.com