← 如何使用 MakeCert 建立開發測試用途的自簽憑證 (Self-Signed Certificate)　　如何在多個 .NET Core SDK 版本之間進行切換 (global.json) →

認識 PKI 架構下的數位憑證格式與憑證格式轉換的心得分享

分享

2018/04/22 00:24　Will 保哥　心得分享, 系統管理

讚　分享　6 人說這個讚。成為朋友中第一個說讚的人。

數位憑證相關的知識真的頗為複雜，以前對這方面的理解都過於片段，上網找到的文章資料就算真的將命令打對了，要嘛沒有講解為什麼，要嘛就是在觀念講解的部分不夠清楚，或是內文描述有許多謬誤之處，以至於每次遇到憑證問題都深感信心不足。我最近又多花了些時間研究，這次對整個數位憑證架構總算有點理解，今天這篇文章主要想來介紹關於憑證格式方面的觀念。

在了解憑證格式之前，建議各位先行 瞭解公開金鑰加密 (Public Key Cryptography) 與 瞭解數位憑證 (Digital Certificates) 的基本觀念，這部分的知識與觀念是幫助你理解 公開金鑰基礎建設 (Public Key Infrastructure) (PKI) 的基石。

簡單來說，PKI 基礎建設包含了：

- A ***certificate authority*** (CA) that stores, issues and signs the digital certificates
  一個 **憑證授權單位** (CA)，用來儲存、發行、簽署數位憑證。
- A ***registration authority*** which verifies the identity of entities requesting their digital certificates to be stored at the CA
  一個 **註冊機構** (RA)，用來驗證申請憑證的實體身分 (個人、法人、應用程式、...)。
- A ***central directory***—i.e., a secure location in which to store and index keys
  一個 **中央目錄**，用來儲存金鑰與檢索金鑰的地方。
- A ***certificate management system*** managing things like the access to stored certificates or the delivery of the certificates to be issued.
  一套 **憑證管理系統**，用來管理像是存取憑證、傳遞已簽發憑證的系統。
- A ***certificate policy*** stating the PKI's requirements concerning its procedures. Its purpose is to allow outsiders to analyze the PKI's trustworthiness.
  一套 **憑證管理政策**，用來規範維護 PKI 基礎架構下所有的管理程序，確保憑證與金鑰的安全性。

在開發與測試的情境下，你只要自己建立一個 **憑證授權單位** (CA) 外加一套 **憑證管理系統** (e.g. OpenSSL)，就可以自行架設一整套 PKI 基礎建設。

基本上，要建立一個 **憑證授權單位** (CA)，首先你必須要先建立一個 **私密金鑰** (Private key)，用來對未來要簽發的 **憑證** (Certificate) 進行 **簽章** (Signature)。這部分你可以使用的管理工具很多，在 Windows 平台，你可能會用 MakeCert 或 PowerShell 來建立 CA 所需的**私密金鑰與憑證**；而在 Linux 平台，大部分人都會選擇採用複雜的 OpenSSL 工具，這套工具參數與選項全部加起來應該有數百種組合，功能強大。

1. 使用 MakeCert 工具建立 CA 的命令如下

這個 MakeCert 工具會隨著 Visual Studio 或 Windows SDK 安裝的過程自動安裝，主要可以產生**僅供測試用途**的 X.509 憑證 (RFC5280)。

```
makecert -n "CN=Will Certificate Authority" -cy authority -a sha1 -sv "CA-PrivateKey.pvk" -r "CA-Certificate.cer"
```

2. 使用 OpenSSL 工具建立 CA 的命令如下

這個 OpenSSL 則是跨平台的 TLS/SSL 工具組，可以用來建立、簽發、轉換憑證格式等多功能用途。

```
openssl genrsa -des3 -out rootCA.key 4096
openssl req -x509 -new -nodes -key rootCA.key -sha256 -days 3650 -out rootCA.crt
```

從上述兩個工具所產生的**私密金鑰與憑證**你就會發現，他們之間的檔案內容格式完全不一樣，MakeCert 所建立的**私密金鑰與憑證是二進位格式**；而 OpenSSL 所建立的**私密金鑰與憑證**預設為文字格式。但事實上，不管你用什麼工具產生**私密金鑰與憑證**，都在做一樣的事情，只是格式不同而已，而且不同格式之間也可以互相轉換。

接著，我們就要進入重點了，介紹這些**私密金鑰與憑證**檔案的格式種類。

一般來說，跨平台共通標準的編碼格式有兩種：

1. **DER (Distinguished Encoding Rules)**
   - 檔案內容為**二進位格式**
   - DER 是 ASN.1 語法標準下的其中一種二進位的編碼方法
   - 常見副檔名有
     - `*.der`
     - `*.cer` (常用於Windows作業系統)
     - `*.crt` (這種副檔名比較不好猜格式，建議少用)
2. **PEM (Privacy Enhanced Mail)**
   - 檔案內容為**文字格式**
   - 其內容是 DER 檔案內容經由 Base64 編碼過後的字串
   - 以 `-----BEGIN ***-----` 開頭，以 `-----END ***-----` 結尾
   - 詳見 RFC 1421、RFC 1422、RFC 1423、RFC 1424
   - 常見副檔名有
     - `*.pem`
     - `*.cert` (常用於Linux作業系統)
     - `*.crt` (這種副檔名比較不好猜格式，建議少用)

> **備註**：CRT 與 CER 或 CERT 都是 Certficiate (憑證) 這個單字的縮寫。

所以無論你想要建立的是**私密金鑰**或是**憑證**，都可以自由選擇想要儲存的格式。但是在這三十多年的數位憑證發展過程中，出現了許許多多不同的規格，用著各種不同的格式來保存**私密金鑰**或**憑證**，以至於 OpenSSL 變得這麼複雜，參數與選項都多到爆炸，任何一個初學者想好好掌握工具使用，大多會被排山倒海的專有名詞給嚇跑。

以下我就來介紹幾個不同的檔案格式：

- **KEY**
  - 用來存放一個**公鑰**或者**私鑰**，這種檔案並沒有事先定義格式，只是意義上的「**私密金鑰**」而已。
  - 編碼格式可能是 **PEM** 或 **DER**，但通常是 **PEM** 格式 (因為在 Linux 或 Node.js 環境下很常用)
  - 金鑰內容可能是加密過的，但也可能是未加密過的，要依據實際內容而定。
  - 常見副檔名有
    - `*.key`
- **PVK (Microsoft PrivateKey Blob)**
  - 微軟專屬的「**私密金鑰**」格式
  - PVK 是二進位檔案，檔案內容格式可參見：PVK file format
  - 透過 MakeCert 所產生的**私密金鑰**，預設就是 **PVK** 格式
  - 這種檔案只有「**私密金鑰**」，不會包含「**憑證**」在內
  - PVK 可以透過 OpenSSL 工具轉換為 PEM 格式，命令如下：
    - `openssl rsa -in "ca.pvk" -inform PVK -out ca.key -outform PEM`
  - PVK 也可以轉換為 PFX 格式，透過 WDK 內建的 Pvk2Pfx 工具，就可以進行轉換：
    - `"C:\Program Files (x86)\Windows Kits\10\bin\10.0.16299.0\x64\pvk2pfx.exe" -pvk "server.pvk" -spc "server.cer" -pfx "server.pfx"`
    - 備註：通常 WDK (Windows Driver Kit) 會隨著 Visual Studio 安裝過程自動安裝好。
  - 常見副檔名有
    - `*.pvk`
- **CSR (Certificate Signing Request) / PKCS #10**
  - CSR = 憑證簽署請求檔
  - 當你需要讓 CA 幫你簽署一份憑證時，通常要先準備一把 **私密金鑰** (Private Key)，然後先產生 CSR 檔案，這份 CSR 檔案將包含 **主體資訊** (Subject) 與一把 **公開金鑰** (Public Key)，然後將 CSR 檔案提交給 CA 進行簽署，之後就可以由 CA 簽署過，並產生一份含有 **公開金鑰** (Public Key) 的 **憑證** (Certificate) 回來。
  - **主體資訊** (Subject) 通常包括憑證的摘要資訊，例如國家代號、省分、城市、組織名稱、單位名稱、一般名稱 (Common Name) 等等，也可以外加一些 **擴充屬性** ('extra' attributes) 供 CA 簽署時參考。
  - 詳見 RFC 2986 - PKCS #10: Certification Request Syntax Specification
  - 你可以透過 OpenSSL 工具建立一個 CSR 檔案，命令如下：(先擁有私密金鑰，再建立 CSR 檔案)
    - `openssl genrsa -out server.key 2048`
    - `openssl req -new -sha256 -out` **`server.csr`** `-key server.key`
  - 你可以透過 OpenSSL 工具查看 CSR 檔案的請求內容，命令如下：
    - `openssl req -noout -text -in` **`server.csr`**
  - 常見副檔名有
    - `*.csr`
- **P7B / PKCS#7**
  - P7B/PKCS#7 檔只會包含**憑證與中繼憑證**，不會包含**私密金鑰**。
  - 詳見 RFC 2315 - PKCS #7: Cryptographic Message Syntax
  - 其內容採用 PEM 編碼 (意即 Base64 編碼過的文字格式)
  - PKCS#7 格式的憑證主要用來對訊息**簽章**或**加解密**。
  - 完整的PKCS參考：https://en.wikipedia.org/wiki/PKCS
  - 必較常見會用到 `P7B` 格式的平台只有 Microsoft Windows 與 Java Tomcat
  - 常見副檔名有
    - `*.p7b`
    - `*.p7c`
    - `*.keystore (在 Java 開發環境下常用這個副檔名)`
- **PKCS#8**
  - PKCS#8 檔主要包含**私密金鑰**，且這份私密金鑰**必須設定密碼**！
  - 詳見 RFC 5958 - Asymmetric Key Packages 與 RFC 5208 - Public-Key Cryptography Standards (PKCS) #8: Private-Key Information Syntax Specification Version 1.2
  - 其內容採用 PEM 編碼 (意即 Base64 編碼過的文字格式)
    - 檔案開頭: `-----BEGIN ENCRYPTED PRIVATE KEY-----`
    - 檔案結尾: `-----END ENCRYPTED PRIVATE KEY-----`
  - 你可以透過 OpenSSL 工具將一個 **RSA 加密過的私密金鑰**轉換為 **PKCS#8 密碼保護的金鑰**版本 (一樣為 PEM 格式)，命令如下：
    - `openssl pkcs8 -in server.key -out server.pkcs8.key -topk8`
  - 你可以透過 OpenSSL 工具將一個 **PKCS#8 密碼保護的金鑰**轉換為**未加密金鑰**版本 (一樣為 PEM 格式)，命令如下：
    - `openssl pkcs8 -in server.pkcs8.pem -out server.traditional.key -topk8`
  - 常見副檔名有
    - `*.key`
    - `*.pkcs8.key`
- **PFX / PKCS#12 (predecessor of PKCS#12)**
  - 在 Linux 平台下，通常**憑證檔**與**金鑰檔**都是分開保存的，所以當在設定網站 TLS/SSL 時，通常會需要設定兩個檔案。
  - 在 Windows 平台下，IIS 改用 PFX 檔案格式，將**憑證**與**金鑰**結合在一個檔案裡，並加入一些擴充屬性(Metadata)。
  - 後來 PKCS 推出一份 PKCS#12 規格 (二進位格式)，可以在一個檔案中同時包含**憑證**、**中繼憑證**、**私密金鑰**，取代 PFX 成為標準格式。
  - 詳見 RFC 7292 - PKCS #12: Personal Information Exchange Syntax v1.1
  - 產生 `*.pfx` 檔案通常需要一組密碼，保護 PFX 檔案的安全性。
  - 你隨時可以將 `*.pfx` 檔案轉換成 PEM 編碼格式 (檔案中將會包含所有憑證與金鑰的 PEM 格式內容)，OpenSSL 的指令如下：
    - `openssl pkcs12 -in` **`mysite.pfx`** `-out` **`mysite.pem`** `-nodes`
    - 備註：上述命令的 `-nodes` 參數，其實是 No DES 加密的意思，也就是產生出來的私密金鑰不需要加密。
  - 產生 PFX 檔案的 OpenSSL 命令如下：
    - `openssl pkcs12 -export -in server.crt -inkey server.key -out server.pfx`
    - `openssl pkcs12 -export -in server.crt -inkey server.key -out server.pfx -certfile CACert.crt`
    - 備註：上述命令的 `-certfile` 用來將額外的憑證一併加入 PFX 檔案，通常是一個以上的**中繼憑證**或**根憑證**。
  - 常見副檔名有
    - `*.pfx`
    - `*.p12`
    - `*.pkcs12`
- **JKS (Java Key Storage)**
  - 這是一種 Java 專用的金鑰格式，有專利保護 (我在設定 Jenkins 的時候有看過)
  - 你可以利用 Java的 **`keytool`** 工具來產生 `*.jks` 檔案
  - 透過 Java 的 **`keytool`** 工具也可以將 PFX 轉為 JKS 格式
  - 詳見 keytool-Key and Certificate Management Tool
  - 常見附檔名有
    - `*.jks`

終於介紹完了到目前為止我所知道的金鑰與憑證格式，當然在 Wikipedia 的 PKCS (Public Key Cryptography Standards) 文件中，你還會看到更多其他不同時期定義出來的規格，有需要再去查看了解即可。

相關連結

- 新手上路
  - Understanding Public Key Cryptograph
  - Understanding Digital Certificates
  - Understanding Certificates and PKI - Technical Documentation - Support - Juniper Networks
  - Public key infrastructure - Wikipedia
  - 紐菲斯的部落格 » X.509 的基礎觀念
  - PKCS - Wikipedia
  - certificate - What is a Pem file and how does it differ from other OpenSSL Generated Key File Formats? - Server Fault
  - .NET Framework Cryptography Model | Microsoft Docs
- MakeCert
  - MakeCert | Microsoft Docs
  - 產生並匯出點對站的憑證：MakeCert：Azure | Microsoft Docs
  - [研究] makecert.exe 憑證建立工具 安裝
- OpenSSL
  - openssl-rsa, rsa - RSA key processing tool
  - openssl-genrsa, genrsa - generate an RSA private key
  - openssl-req, req - PKCS#10 certificate request and certificate generating utility
  - openssl-x509, x509 - Certificate display and signing utility
  - openssl-pkcs8, pkcs8 - PKCS#8 format private key conversion tool
  - openssl-pkcs12, pkcs12 - PKCS#12 file utility

標籤：PKI, 憑證, 格式轉換, CA, certificate

相關文章

認識 PKI 架構下的數位憑證格式與憑證格式轉換的心得分享
數位憑證相關的知識真的頗為複雜，以前對這方面的理解都過於片段，上網找到的文章資料就算真的將命令打對了，要嘛沒有講解...

認識 Angular Library 函式庫專案並學會自製 Angular 表單驗證器模組
當 Angular 越用越熟，你將會發現其開發效率極高無比，除了極佳的工具支援外，優異的模組化技術更是不在話下。上週在教 Ang...

認識 Azure Web App 內建的 Kudu 引擎
很多人可能不知道 Azure Web App (之前叫做 Azure Web Sites) 背後有個強大的管理工具叫做 "Kudu"，這個 Kudu 引擎可以用來...

2 Comments　The Will Will Web　Disqus' Privacy Policy　1 Login

Favorite 3　Tweet　Share　Sort by Best

Join the discussion…

LOG IN WITH　OR SIGN UP WITH DISQUS ?

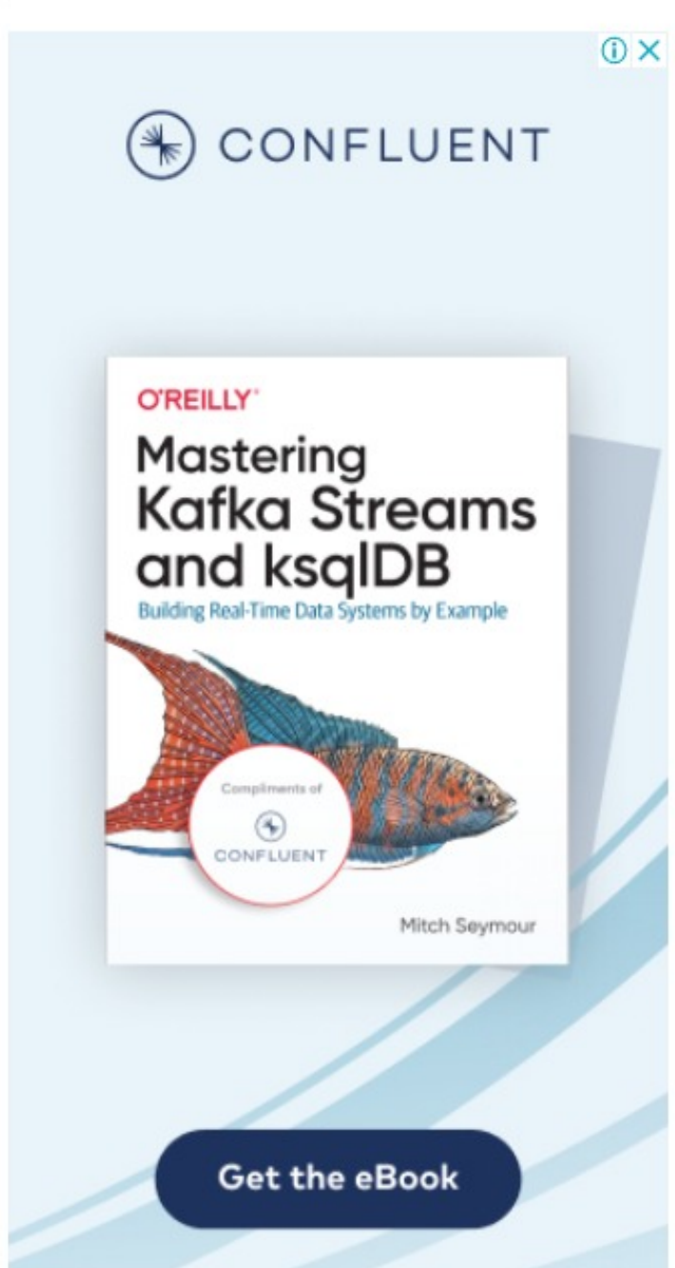
Name

YuTse Chien · 4 years ago
我有時會用 OpenSSL的 asn1parse 來看key
因為拿到key有時沒有開頭跟結尾
1 · Reply · Share ›

Will Mod → YuTse Chien · 4 years ago
asn1parse 可以看所有合法的 ASN.1 結構，輸出的資料是醜了點，但是內容相當完整。
多謝分享 ^_^
Reply · Share ›

Subscribe　Add Disqus to your site　Do Not Sell My Data　DISQUS

工商服務 (廣告)



搜尋...　搜尋

每月文章
2022
三月 (7)
二月 (9)
一月 (15)
2021
2020
2019
2018
2017
2016
2015
2014
2013
2012
2011
2010
2009
2008
2007

文章分類
.Net (215)
.NET Core (53)
Accessibility (3)
Angular (22)
AngularJS (11)
ASP.NET (222)
ASP.NET 5 (3)
ASP.NET Blazor (1)
ASP.NET Core (31)
ASP.NET Identity (2)
ASP.NET MVC (104)
ASP.NET Web API (13)
Azure DevOps (24)
C# (124)
Cloud Computing (4)
CSS (29)
DevOps (20)
Docker (23)
Entity Framework (21)
Git (28)
Golang (2)
HTML5 (8)
IIS (103)
Java (7)
JavaScript (108)
Jenkins (7)
Kubernetes (9)
LINQ (36)
Linux (108)
Microsoft Azure (35)
MySQL (15)
Office (49)
Office 365 (28)
Oracle (10)
PHP (25)
Scrum (1)
Security (60)
SQL Server (125)
Subversion (35)
System Center (2)
TFS (6)
TFS2010 (10)
Tips (188)
Unit Testing (10)
Usability (1)
VBA (5)
Visual Basic (5)
Visual Studio (122)
Visual Studio 11 (2)
Visual Studio 2012 (10)
Visual Studio 2013 (4)
VS2010 Tips (23)
Web (167)
WebMatrix (8)
Windows (38)
Windows 8 (14)
Windows Azure (18)
Windows Phone 7 (14)
介紹好用工具 (221)
心得分享 (85)
多奇快訊 (7)
系統管理 (358)
前端工程研究 (18)
專案管理 (8)
團隊合作 (9)
網路管理 (18)

專業證照

Microsoft CERTIFIED Professional Developer　Web Developer 4
Microsoft CERTIFIED Technology Specialist　.NET Framework 4, Web Applications

聯播文章

黑暗執行緒
2022 香魚馬
Git 實戰 - 將檔案從歷史 Commit 中移除
【茶包射手筆記】AD Domain Controller 出現某主機的本機帳號登入錯誤

Scott Hanselman
Using Home Assistant to integrate a Unifi Pro...
JavaScript and TypeScript Projects with React...
A Nightscout Segment for OhMyPosh shows my re...

Download OPML file

免責聲明

微軟最有價值專家 & Google 開發專家

Microsoft Most Valuable Professional　Experts 2019 Angular GDE