SSD 測試筆記

SECURE SOFTWARE DOWNLOAD
CVC 應用與製作

Jimmy Huang

February 4, 2016

Contents

1 Introduction

以下內容隨便翻譯, 整合 3.0 和 3.1 的內容

2 Overview

本章節所定義的需求用來達成安全下載軟體的目標

- CM 應該要有方法去驗證將被下載的軟體其來源是可信的的。
- CM 應該要有方法去驗證將被下載的軟體沒有被竄改。
- 這一過程應力求簡化了運營商的軟體檔案處理需求，並提供機制讓運營商可以升級或降級其網絡上的 CM 的軟體。
- 該過程允許運營商決定並相對於控制他們的策略：（a）該代碼的文件將是通過他們的網絡中的 CM 接受; 及（b）定義的過程的安全性的安全控制他們的網絡; The process allows operators to dictate and control their policies with respect to: (a) which code files will be accepted by CMs within their network; and (b) security controls that define the security of the process on their network;
- CM 能在不同的運營商控制的系統之間自由移動
- 支援更新 certiticate
  - DOCSIS 3.0
    * 支援更新存在 CM 中的 Root CA Public Key (optional)
    * 支援更新存在 CM 中的 Manufacturer CA Certificate (optional)
    * 支援更新存在 CM 中的 CableLabs CVC Root CA Certificate (optional)
    * 支援更新存在 CM 中的 CableLabs CVC CA Certificate (optional)

    DOCSIS 3.0 CM 目前沒有使用 CableLabs CVC Root CA Certificate 和 CableLabs CVC CA Certificate，若將來 manufacturer CVC 改由 CableLabs CVC Root CA Certificate chain 簽發才會用到。
  - DOCSIS 3.1
    * 支援更新存在 CM 中的 Root CA Certificate (optional)
    * 支援更新存在 CM 中的 Device CA Certificate (optional)
    * 支援更新存在 CM 中的 legacy Root CA Public Key (optional)
    * 支援更新存在 CM 中的 legacy Manufacturer CA Certificate (optional)

  本文件限制了這些主系統的安全性的要求的範圍，但必須承認，在某些情況下可能需要額外的安全性。個體經營者或 CM 廠商可能會因為這些疑慮而在 image 或是網路上加入額外的安全機制。只要不和本規範衝突，並不禁止這些額外的保護措施

Multiple levels of protection are required to protect and verify the code download:

- CM 軟體的製造商一定會在軟體加上數位簽章。這個數位簽章可透過延伸到 Root CA 的憑證串鍊加以驗證。製造商的數位簽章則用來確認 CM 軟體的完整性
- 雖然製造商一定會在軟體加上數位簽章，運營商可額外加入數位簽章。若出現第二份簽章，則在接受這個軟體之前 CM 要先利用延伸到 Root CA 的憑證串鍊驗證數位簽章
- 正確執行這個過程對 CM 配置和控制的 OSS 機制來說是很重要的。CM 的軟體升級功能在配置和註冊過程中啓用。軟體升級是在配置和註冊程序啓動，或是使用 SNMP 來啓動。

code file 被包裝成 [PKCS#7] 結構，裡面包含

- code image，就是升級用軟體

- Code Verification Signature (CVS)，就是這個結構的數位簽章。
- Code Verification Certificate (CVC)，用來發布驗證 code image 的數位簽章的公鑰的 certificate。簽署這張 certificate 的 DOCSIS Certificate Authority 所用的 public key 存放在 CM 裡。

下圖顯示出只有製造商簽署 code file，以及製造商和運營商都簽署 code file 所需要的基本步驟。在 DOCSIS 中，DOCSIS 3.0 CM 會安裝 DOCSIS Root Certificate Authority 的公鑰。DOCSIS 3.1 CM 會安裝 Root CA certificate。軟體的製造商會用 Mfr CVC 為軟體加上 PKCS#7 格式數位簽章，若是使用 new PKI 會在加上一張 CVC CA certificate。之後把簽署過得 code file 送給運營商。運營商驗證軟體沒有被竄改。這時運營商可以把軟體放到軟體更新用的伺服器，或是以相同方式加上一組運營商的為簽章。在軟體升級的過程中，CM 會在安裝前先驗證軟體是否可以信任。

Currently, the DOCSIS Root CA issues both Manufacturer CA certificates and CVCs. In the future the CableLabs Manufacturer Root CA certificate may be used to issue and validate Manufacturer CA certificates and the CableLabs CVC Root CA may be used to issue and validate CVCs.

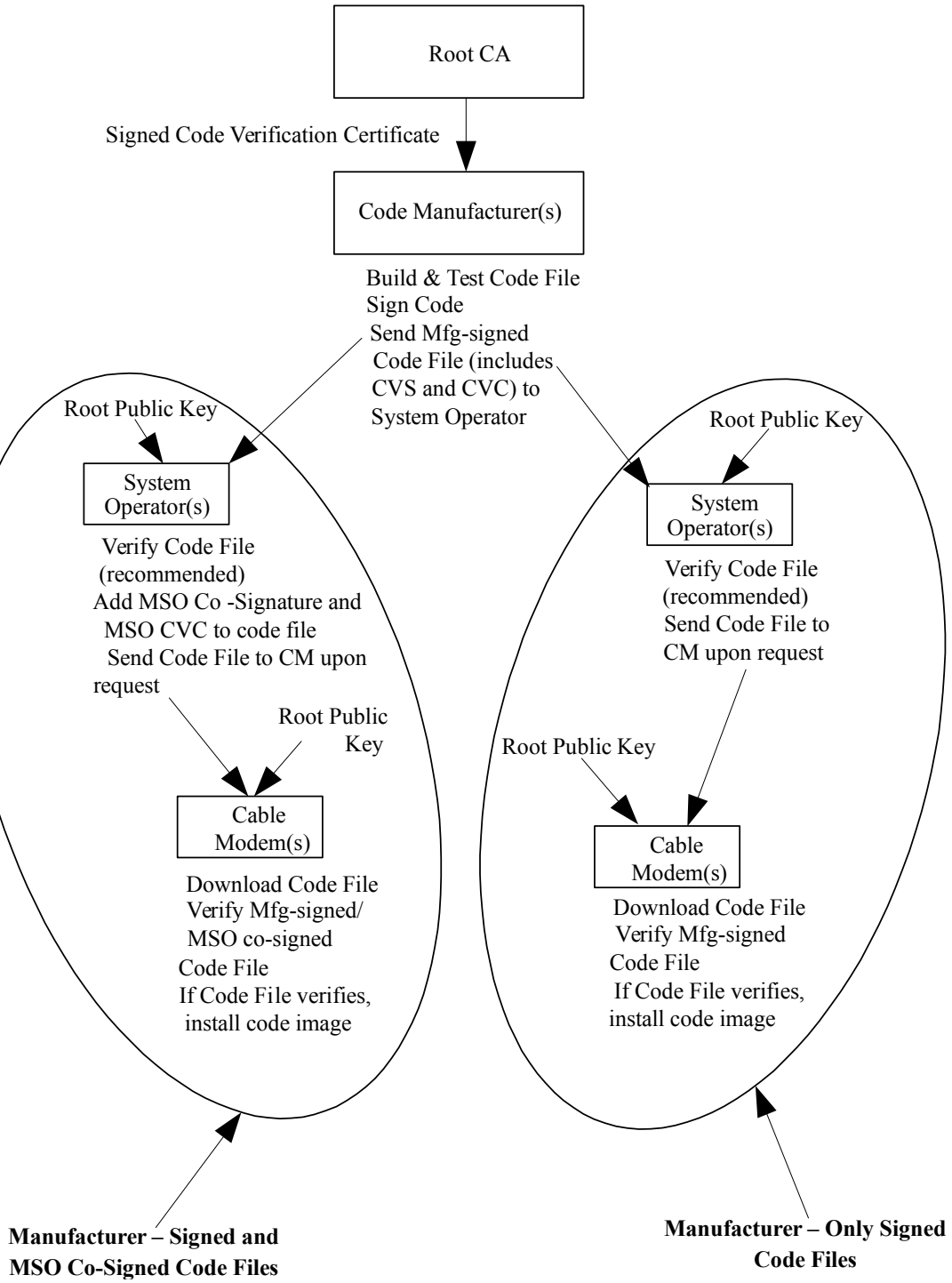

Figure 1: DOCSIS 3.0 Code Validation Hierarchy

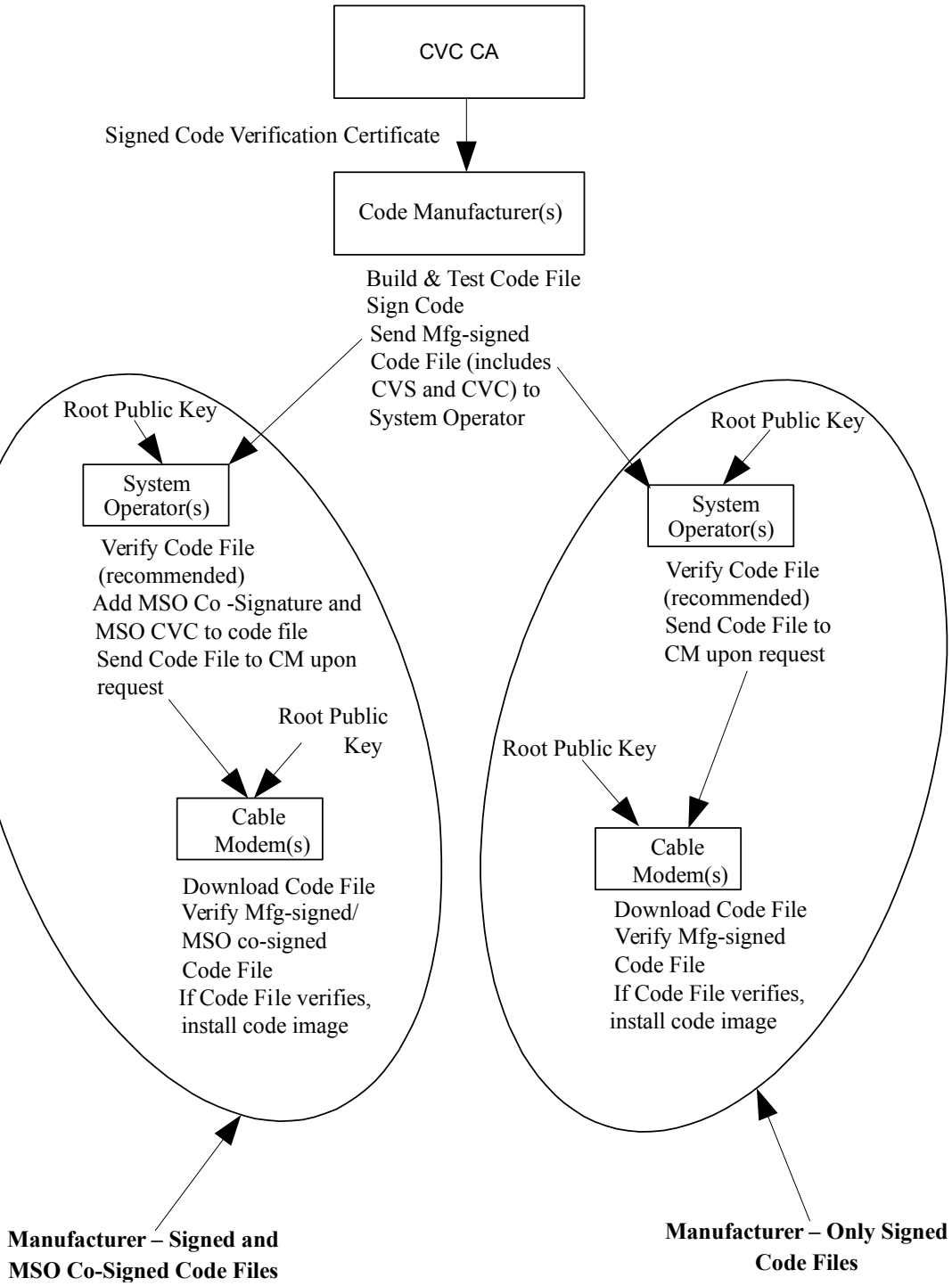

Figure 2: DOCSIS 3.1 Code Validation Hierarchy

3 Software Code Upgrade Requirements

The following sections define requirements of the CM software code upgrade verification process.

DOCSIS 3.0 CM

所有 DOCSIS 1.1 2.0 3.0 的軟體都要能被驗證。所有 DOCSIS 1.1 2.0 3.0 CM **必須**驗證升級軟體，不論 BPI 是否開啓

DOCSIS 3.1 CM

所有 DOCSIS 3.1 的軟體都要能被驗證。所有 DOCSIS 3.1 CM **必須**驗證升級軟體，不論 BPI 是否開啓用來發布 new PKI CVC 的 certificate 有三種：Root CA，CVC CA，以及 CVC。CableLabs 控管 new PKI 以及由它的 CAs (CableLabs Root CA and CableLabs CVC CA) 所簽發的 certificate。CM **必須**使用 RFC5280 的方式處理 CVC extensions。為了向下相容，CM **必須**支援使用 legacy PKI 的方式升級軟體。

3.1 Code File Processing Requirements

The code file format is defined in Appendix III. CM **必須**拒用 code file The CM MUST reject the DOCSIS [PKCS#7] code file if the signedData field does not match the DER encoded structure represented in Appendix III. CM **必須**能夠驗證金鑰長度為 1024，1536，2048 bits 的數位簽章。公鑰的 exponent 為 F_4 (65537) [1] CM **必須**拒絕不符合 Appendix III 規定的 DER 編碼的 CVC。除非已經驗證過，否則 CM **必須不**安裝升級用的軟體。若軟體下載安裝成功且 code file 內含有 DOCSIS Root CA Public key (type 4)，CM **必須**用這個值取代目前存在 CM 的值。若軟體下載安裝成功且 code file 內含有 Root CA Certificate (type 54)，CM **必須**用這個值取代目前存在 CM 的值。若軟體下載安裝成功且 code file 內含有 Manufacturer CA certificate (type 17)，CM **必須**用這個值取代目前存在 CM 的值。若軟體下載安裝成功且 code file 內含有 CVC Root CA 及 CVC CA certificate，CM **必須**把這兩張存起來以便將來能 SSD。

3.2 Code File Access Controls

除了利用 digital signature 與 certificate 所提供的加密控制之外，code file 裡還有一些特殊的值讓 CM 在確認 code image 有效之前可以檢查。在 CM 嘗試驗證 CVC 和 CVS 之前，**必須**先符合這些值所設定的條件。

3.2.1 Subject Organization Names

CM **必須**能辨識出 2 個出現在 CVC 的 subject field 的名稱，這些名稱被視為可信任的 code-signing agent

- The cable modem manufacturer：

  CM **必須**驗證 manufacturer CVC 的 subject field 的 manufacturer name 與存在 CM non-volatile memory 裡的 manufacturer name 是完全一致的。在 code file 中一定會包含 manufacturer CVC。在 DOCSIS 3.1 CM 中**必須**在 non-volatile memory 中分別除存儲存 legacy PKI 和 new PKI 的 manufacturer name。

- A co-signing agent

  如上所述，DOCSIS 允許另一個受信任的單位協同簽署軟體。這個單位最常見的就是運營商。這個 co-signing agent 的 organization name 會在 CM 初始軟體驗證過程中透過 configuration file 的 co-signer CVC 傳遞給 CM。CM **必須**驗證 co-signer CVC 中的 co-signer organization name 與上一次透過這個流程取得的 organization name 一致，並把這個值存到 CM 中。

3.2.2 Time Varying Controls

為了支持軟體升級的流程，CM **必須**存有 2 組與 code-signing agent 相關的 UTC time。這 2 組值為 codeAccessStart 與 cvcAccessStart。CM **必須**存儲和維護一組 manufacturer signing agent 的 UTC time。若有用到 co-signing agent，CM **必須**存儲和維護一組 co-signing agent 的 UTC time。藉由控制 CVS 與 CVC，這些值能用來控管 CM 對 code file 的存取。存在 CM 中的 UTC time 精準度 **必須**到秒，能表示的時間範圍為 1950 年 1 月 1 日子夜到 2050 年 1 月 1 日子夜。CM **必須不**讓 manufacturer signing agent 的 codeAccessStart 和 cvcAccessStart 減少。若 co-signing agent 沒有改變，CM **必須不**讓 co-signing agent 的 codeAccessStart 和 cvcAccessStart 減少。DOCSIS 3.1 CM 針對 manufacturer code signing agent **必須**分別為 legacy PKI 與 new PKI 各儲存一組 UTC time。

[1] $F_4 = 2^{2^4} + 1$

3.3 Cable Modem Code Upgrade Initialization

在 cable modem 升級之前要先被正確地初始化。Its manufacturer 先初始化 cable modem。之後每次 cable modem 上線 **必須**檢查目前的初始化狀態 with respect to 特定網絡的運營需求 cable modem 可能在註冊時需要被重新初始化，特別是 cable modem 換到其他網路的時候。

3.3.1 Manufacturer Initialization

In support of code upgrade verification, values for the following parameters MUST be loaded into the CM's non-volatile memory: 為了能夠升級，以下參數必須被存在 CM flash。

- CM manufacturer's organizationName
- codeAccessStart initialization value
- cvcAccessStart initialization value

CM **必須**初始 codeAccessStart 和 cvcAccessStart 的值等於 manufacturer 最新的 CVC 的有效起始時間。這些值將會透過 cable modem 接收與驗證過的 manufacturer's CVCs 而被定期更新。

3.3.2 Network Initialization

CM 會從 configuration file 收到和驗證升級相關設定。在 CM 成功註冊上 CMTS 之前 **必須**使用這些設定。configuration file 通常包括適用於 CM 最新的 CVC。當 configuration file 被來啓動升級，它會包含 CVC 初始化 CM，讓 CM 能夠升級。不管是否需要升級，CM 都 **必須**處理在 configuration file 中的 CVC。

configuration file 可能會以下 CVC 組合

DOCSIS 3.0 之前，或是 DOCSIS 3.1 使用 legacy PKI

- No CVC
- A Manufacturer CVC (Type 32)
- A Co-signer CVC (Type 33);
- Both Manufacturer CVC and Co-signer CVC

DOCSIS 3.1 使用 DOCSIS 3.1 PKI

- No CVC
- A Manufacturer CVC Chain (the Manufacturer CVC and its issuing CA certificate (Type 81))
- A Co-signer CVC Chain (the Co-signer CVC and its issuing CA certificate (Type 82))
- Both Manufacturer CVC and Co-signer CVC

在 CM 啓用軟體升級功能之前，**必須**先在 configuration file 中收到有效的 CVC 並且成功註上測 CMTS。假如 CM 的 configuration file 中沒有有效的 CVC 且其軟體升級功能被禁用，則 CM **必須**拒絕任何從 SNMP 方式收到的 CVC 訊息。

若 CM 的 configuration file 不包含 co-signer CVC，則 CM **必須不**接受被 co-signing agent 簽署過的 code files。假如 CM 被設定成要接受 co-signing agent 簽署過的 code files，則當 CM 處理過 co-signer CVC 後**必須**把以下參數存到 memory 中。

- co-signing agent's organizationName
- co-signer cvcAccessStart
- co-signer codeAccessStart

和 manufacturer organizationName 與其 time varying control values 不同，co-signer organizationName 與其 time varying control values 不要求要存到 non-volatile memory.

3.3.2.1 Processing the Configuration File CVC

當 configuration file 有包含 CVC 時，CM 在接受任何 configuration file 中與軟體升級相關的設定時**必須**先驗證 CVC。當收到 CVC 時 CM **必須**執行以下驗證步驟。若以下任何步驟檢驗失敗，CM **必須**立刻停止 CVC 驗證過程。

假如 CM configuration file 不包含有效的 CVC，CM **必須不**下載升級用的 code files，不論是由 configuration file 或是 SNMP 觸發的軟體升級。假如 CM configuration file 不包含有效的 CVC，CM **應該不**處理之後由 SNMP 收到的 CVC。假如 CM configuration file 不包含有效的 CVC，CM **必須不**接受之後由 SNMP 收到的 CVC 的資訊。

在收到含有 CVC 的 configuration file，並且 CM 成功註冊上 CMTS，CM **必須**

1. 驗證 CVC 的 extension 有 Extended Key Usage 值為 codeSigning (1.3.6.1.5.5.7.3.3) 標示為 critical。
2. 若是 manufacturer's CVC，且其 subject organizationName 與 CM 的 manufacturer name 相同，驗證 CVC 的有效起始時間大於等於目前 CM 的 manufacturer's cvcAccessStart value。
3. 若是 manufacturer's CVC，且其 subject organizationName 與 CM 的 manufacturer name 不同，要拒絕這張 CVC 紀錄錯誤訊息。
4. 若是 Co-signer's CVC，且其 subject organizationName 與 CM 目前的 code co-signing agent 相同，驗證 CVC 的有效起始時間大於等於目前 CM 的 co-signer's cvcAccessStart value。
5. 若是 Co-signer's CVC，且其 subject organizationName 與 CM 目前的 code co-signing agent 不同，在驗證完 CVC 且註冊上後，把 CM 的 code co-signing agent 更新成 CVC 的 subject organization name。
6. 驗證 certificate signature。若是 Legcy-PKI，利用存在 CM 裡的 DOCSIS Root CA public key 驗證 CVC。若是 New-PKI，驗證 CVC CA 以及存在 CM 的 Root CA Certificate 組成憑證串鍊。
7. 若是使用 new PKI，驗證 CVC 和 CVCCA 還沒過期。
8. 用驗證過的 CVC 的有效起始時間更新目前 CM 相對應的 cvcAccessStart (manufacturer 或 code co-signing)。假如 CVC 有效起始時間大於 CM 目前的 codeAccessStart，把 codeAccessStart 更新成 CVC 有效起始時間。

3.3.2.2 Processing the SNMP CVC

當 CM 開啓軟體升級功能時，**必須**處理從 SNMP 收到的 CVCs。若 CM 沒開啓軟體升級功能時，**必須**拒絕所有從 SNMP 收到的 CVCs。從 SNMP 收到的 CVC 將利用和驗證 configuration file CVCs 相同的 Root CA certificate 或 public key 去驗證。當收到來自 SNMP 的 CVC，CM **必須**依以下步驟驗證。若以下任何步驟檢驗失敗，CM **必須**立刻停止 CVC 驗證過程。

當 CM 收到來自 SNMP 的 CVC，CM **必須**

1. 驗證 CVC 的 extension 有 Extended Key Usage 值為 codeSigning (1.3.6.1.5.5.7.3.3) 標示為 critical。
2. 若收到的 CVC subject organizationName 與 CM 目前的 co-signing agent 相同，驗證 CVC 的有效起始時間大於目前 CM 的 co-signer cvcAccessStart。
3. 若收到的 CVC subject organizationName 與 CM 的 manufacturer name 相同，驗證 CVC 的有效起始時間大於目前 CM 的 manufacturer's cvcAccessStart value。
4. 拒絕 subject organizationName 不等於 CM 的 manufacturer name 或目前的 co-signing agent name 的 CVC。
5. 驗證 certificate signature。若是 Legcy-PKI，利用存在 CM 裡的 DOCSIS Root CA public key 驗證 CVC。若是 New-PKI，驗證 CVC CA 以及存在 CM 的 Root CA Certificate 組成憑證串鍊。
6. 若是使用 new PKI，驗證 CVC 和 CVCCA 還沒過期。
7. 若 CVC 有效起始時間大於 CM 目前的 codeAccessStart，VM **必須**把 codeAccessStart 更新成 CVC 有效起始時間。

3.4 Code Signing Guidelines

3.5 Code Verification Requirements

除非滿足以下驗證，否則 CM **必須不**安裝升級的軟體

3.5.1 Cable Modem Code Verification Steps

CM 下載升級軟體時，**必須**執行以下的驗證。以下流程可一任意順序執行若任一項驗證失敗或是檔案格式不符，CM **必須**立即停止下載的流程並在可行的情況下紀錄錯誤訊息，移除所有執行到該步驟剩餘的流程，並繼續使用目前的軟體運作。

1. The CM MUST verify that:
   - signingTime 大於等於 CM 目前的 manufacturer codeAccessStart。
   - signingTime 大於等於 manufacturer CVC 有效起始時間。
   - signingTime 小於等於 manufacturer CVC 有效結束時間。
2. The CM MUST verify that:
   - manufacturer CVC 的 subject organizationName 要與 CM 所存放的 manufacturer name 相同。
   - manufacturer CVC 有效起始時間大於等於 CM 目前的 manufacturer cvcAccessStart
   - Manufacturer CVC 的 Extended Key Usage 值為 codeSigning (1.3.6.1.5.5.7.3.3) 標示為 critical。
3. 驗證 certificate signature。
   - 若是 Legcy-PKI，利用存在 CM 裡的 DOCSIS Root CA public key 驗證 CVC。
   - 若是 New-PKI，驗證 Mfr CVC 與存在 CM 的 Root CA Certificate 能組成憑證串鍊。
4. If the CVC is issued from the new PKI, verify that the validity periods for the CVC and the issuing CA certificate have not expired.
5. CM **必須**驗證 manufacturer 的 code file signature。若驗證不過，CM **必須**拒絕所有 code file 裡的元件，也應該立即捨棄所有在驗證過程中所得到的值。
6. 若 manufacturer signature 與 co-signing agent signature 要被驗證
   (a) The CM MUST verify that:
      i. code file 包含 co-signer's signature
      ii. signingTime 大於等於 CM 目前相對應的 codeAccessStart
      iii. signingTime 大於等於相對應的 CVC 的有效起始時間
      iv. signingTime 小於等於相對應的 CVC 的有效結束時間
   (b) The CM MUST verify that:
      i. CVC subject organizationName 與 CM 目前的 co-signer's organization name 相同
      ii. CVC 有效起始間大於等於 CM 目前相對應的 cvcAccessStart
      iii. CVC 的 Extended Key Usage 值為 codeSigning (1.3.6.1.5.5.7.3.3) 標示為 critical
   (c) 驗證 certificate signature。
      - 若是 Legcy-PKI，利用存在 CM 裡的 DOCSIS Root CA public key 驗證 CVC。
      - 若是 New-PKI，驗證 Mfr CVC 與存在 CM 的 Root CA Certificate 能組成憑證串鍊。
   (d) If the CVC is issued from the new PKI, verify that the validity periods for the CVC and the issuing CA certificate have not expired
   (e) CM **必須**驗證 co-signer 的 code file signature。若驗證不過，CM **必須**拒絕所有 code file 裡的元件，也應該立即捨棄所有在驗證過程中所得到的值。

7. 一旦 manufacturer (以及 co-signer) 的 signature 被驗證過後，code image 就能夠被信任並進行安裝。在安裝之前，所有在驗證過程中所得到的值，除了 [PKCS#7] signingTime 與 CVC 有效起始時間以外都**應該**立即捨去。
8. CM 可以用 MULPI 規範的方式安裝 code file。
9. 若安裝失敗，CM **必須**捨棄從 code file 中得到的 [PKCS#7] signingTime 與 CVC 有效起始時間。處理這種失敗的流程規範在 MULPI。
10. 一旦成功安裝軟裡後，CM **必須**
    (a) 將目前的 manufacturer codeAccessStart 更新成 [PKCS#7] signingTime
    (b) 將目前的 manufacturer cvcAccessStart 更新成 CVC 有效起始時間。
11. 若成功安裝軟裡，而且是有 co-signing agent 簽署過得 code file，CM **必須**
    (a) 將目前的 co-signer’s codeAccessStart 更新成 [PKCS#7] signingTime
    (b) 將目前的 co-signer’s cvcAccessStart 更新成 CVC 有效起始時間。

3.6 DOCSIS Interoperability

DOCSIS 3.0 CM

DOCSIS 3.0 cable modems MUST verify code upgrades according to this specification even when operating with a DOCSIS environment prior to DOCSIS 3.0. DOCSIS 1.0 configuration files intended for DOCSIS 3.0 cable modems need to support the configuration file requirements that are defined in this specification in order for code upgrades to work properly for a DOCSIS 3.0 CM operating in DOCSIS 1.0 mode.

DOCSIS 3.1 CM

要使用 DOCSIS 3.1 SSD 的 image 要用 New-PKI certificate 簽署。要使用 legcy SSD 的 image 要用 legcy-PKI certificate 簽署。CM 支援使用 new PKI 或 legcy PKI 做 SSD。CM 根據 configuration file 決定使用哪一個 PKI。若 configuration file 內包含 Manufacturer CVC Chain 或 a Co-signer CVC Chain，CM **必須**使用 DOCSIS 3.1 SSD，不論是否有 legacy Manufacturer 或 Co-signer CVC。若 configuration file 內包含 legacy Manufacturer 或 Co-signer CVC 且不包含 Manufacturer CVC Chain 或 a Co-signer CVC Chain，則 CM **必須**使用 legacy SSD。

4 File format

4.1 Certificate

CVC 的格式採用 X.509 標準，使用 DER 編碼，其結構定義於 Table III–2。CVC 包含一個 extension：Extended Key Usage extension。標記為 critical。此 extension 包含一個 KeyPurposeID 其值為 id-kp-codeSigning。若 CVC 不包含 extensionis，或是 extension 沒標記為 critical，或是包含 id-kp-codeSigning 以外的 KeyPurposeID，CM 將會停止驗證流程並捨棄這個 CVC。

Table 1: DOCSIS X.509 Compliant Code Verification Certificate

X.509 Certificate Field	Description
Certificate {	
tbsCertificate	
version	v3(2)
serialNumber	Integer, size (1..20) octets
signature	SHA-1 with RSA, null parameters
issuer	
countryName	US
organizationName	Data Over Cable Service Interface Specifications
organizationalUnitName	Cable Modems
commonName	DOCSIS Cable Modem Root Certificate Authority
validity	
notBefore	utcTime (GMT), YYMMDDhhmmssZ
notAfter	utcTime (GMT), YYMMDDhhmmssZ
subject	
countryName	<country of subject company>
organizationName	<subject code-signing agent>
organizationalUnitName	DOCSIS
commonName	Code Verification Certificate
subjectPublicKeyInfo	
algorithm	RSA encryption, null parameters
subjectPublicKey	1024-bit, 1536-bit, or 2048-bit modulus
extensions	
extKeyUsage	
critical	true
keypurposeId	id-kp-codeSigning
signatureAlgorithm	SHA-1 with RSA, null parameters
signature Value	
} end certificate	

當 DOCSIS Root CA 簽發新的 Manufacturer CVC 時，適用以下條件

- tbsCertificate.validity.notBefore 的值為簽發當下的時間。
- tbsCertificate.validity.notAfter 的值為從簽發當下的時間算起至少 2 年最多 10 年。

在 Manufacturer CVC 過期之前，這張 certificate 要用新的有效時間與序號重新簽發。CM 的廠商需要在 CVC 過期前 6 個月取得尋的 CVC。

當 DOCSIS Root CA 重新簽發 Manufacturer CVC，以下的值將與目前 CVC 相同

- tbsCertificate.issuer
- tbsCertificate.subject

新的起始時間會若在目前 CVC 起始時間與簽發時的時間之間。The tbsCertificate.validity.notBefore value will be between the tbsCertificate.validity.notBefore value of the current DOCSIS Manufacturer CVC, and the actual issuance date and time.

(That is, the tbsCertificate.validity.notBefore value can be the same as the tbsCertificate.validity.notBefore value of the current DOCSIS Manufacturer CVC, the actual issuance date and time, or any value between the two values.) In addition, the tbsCertificate.validity.notAfter will be the actual re-issuance date and time plus 2 to 10 years

DOCSIS Root CA 負責紀錄經授權的 code-signing agents。Code-signing agents 包含製造生與運營商。DOCSIS Root CA 負責保證各個 code-signing agent 的 organizationName 是唯一的。

指派 organizationNames 給 code co-signers 時

- co-signer agent CVC 裡的 organizationName 是由 DOCSIS Root CA 指派的。
- 這個名稱是 8 個十六進制數字的 printable string 用來標示各個不同的 code-signing agent。

十六進制數字的使用範圍是 0x30 到 0x39 0x41 到 0x46 (數字 0-9 大寫 A-F)。0x3030303030303030 保留不用。

Table 2: DOCSIS 3.1 Code Verification Certificate

Version		**v3**		
Serial number		Unique Positive Integer assigned by the CA		
Issuer DN		c=US o=CableLabs ou=CVC CA01 cn=CableLabs CVC Certification Authority		
Subject DN		c=<Country of Manufacturer> o=<Company Name> cn=Code Verification Certificate		
Validity Period		Up to 10 yrs		
Public Key Algorithm		Sha256WithRSAEncryption (1 2 840 113549 1 1 11)		
Keysize		2048-bits		
Parameters		NULL		
Standard Extensions	OID	Include	Criticality	Value
extKeyUsage	id-ce 37	X	TRUE	
codeSigning				Set
authorityKeyIdentifier	id-ce 35	X	FALSE	
keyIdentifier				Calculated per Method 1

4.2 CM Code File Signing Policy

4.3 CM Code File Format

讓 CM 升級用的軟體會封裝成一個檔案。這個檔案是 DOCSIS [PKCS#7] signed data。

若是使用 legacy code file, 其內容會包含：

- The Manufacturer Code Verification Signature (CVS);
- The Manufacturer Code Verification Certificate (CVC), signed by the DOCSIS Root CA;
- The code image (compatible with the destination CM) as signed content;
- Optionally, when the MSO co-signs the code file:
  a) The Co-signer CVS;
  b) The Co-signer CVC signed by the DOCSIS Root CA.
- Optional Root CA Public Key for the CVC verification;
- Optional Manufacturer Certificate(s).

若是使用 DOCSIS 3.1 code file, 其內容會包含：

- The Manufacturer’s Code Verification Signature (CVS);
- The Manufacturer’s Code Verification Certificate (CVC), signed by the DOCSIS Root CA;
- The code image (compatible with the destination CM) as signed content;
- Optionally, when the MSO co-signs the code file:
  a) The MSOs CVS;
  b) The MSOs CVC signed by the DOCSIS Root CA.
- Optional Root CA Public Key for the CVC verification;
- Optional Manufacturer Certificate(s);
- Optional CVC Root CA Certificate;
- Optional CVC CA Certificate.

Code file 格式符合 DER 編碼的 [PKCS#7] 。

Table 3: Legcy CM Code File

Code File Structure	Description
[PKCS#7] Digital Signature{	
ContentInfo	
contentType	SignedData
signedData()	EXPLICIT signed-data content value; includes CVS and [X.509] CVC
}	
SignedContent{	
DownloadParameters {	Mandatory TLV format (Type 28) defined in the Section 7.2.2.28. (Length is zero if there are no sub-TLVs.)
RootCAPublicKey()	Optional TLV for the legacy PKI Root CA Public Key for CVC Verification, formatted according to the RSA-Public-Key TLV format (Type 4) defined in the Section 7.2.2.4.
MfgCert()	Optional TLV for one DER-encoded legacy PKI Manufacturer CA Certificate formatted according to the CA-Certificate TLV format (Type 17) defined in the Section 7.2.2.17.
ClabCVCRootCACert()	Optional TLV for one DER-encoded certificate formatted according to the CVC-Root-CA- Certificate TLV format (Type 51) defined in the section 7.2.2.29.
ClabCVCCACertificate()	Optional TLV for one DER-encoded certificate formatted according to the CVC-CA-Certificate TLV format (Type 52) defined in the section 7.2.2.30.
}	
CodeImage()	Upgrade code image
}	

Table 4: DOCSIS 3.1 CM Code File

Code File Structure	Description
[PKCS#7] Digital Signature{	
ContentInfo	
contentType	SignedData
signedData()	EXPLICIT signed-data content value; includes CVS and [X.509] CVC
}	
SignedContent{	
DownloadParameters {	Mandatory TLV format (Type 28) defined in the Section 7.2.2.28. (Length is zero if there are no sub-TLVs.)
RootCAPublicKey()	Optional TLV for the legacy PKI Root CA Public Key for CVC Verification, formatted according to the RSA-Public-Key TLV format (Type 4) defined in the Section 7.2.2.4.
MfgCert()	Optional TLV for one DER-encoded legacy PKI Manufacturer CA Certificate formatted according to the CA-Certificate TLV format (Type 17) defined in the Section 7.2.2.17.
DeviceCACert()	Optional TLV for one DER-encoded new PKI certificate formatted according to the Device-CA-Certificate TLV format (Type 53) defined in the Section 7.2.2.31.
RootCACert()	Optional TLV for one DER-encoded new PKI certificate formatted according to the Root- CA-Certificate TLV format (Type 54) defined in the Section 7.2.2.32.
}	
CodeImage()	Upgrade code image
}	

4.3.1 DOCSIS PKCS#7 Signed Data

DOCSIS PKCS#7 數位簽章的 signedData 採用 DER 編碼，格式如 Table n 。5 on page 14

The signedData field of the DOCSIS [PKCS#7] Digital Signature matches the DER encoded structure defined in Appendix III.

4.3.1.1 Code Signing Keys

數位簽章採用的是 SHA-1 (Legcy) 或是 SHA-256 (DOCSIS 3.1) 的雜湊配上 RSA 加密。RSA 金鑰長度在 DOCSIS 3.0 以前是 1024 bits，1536 bits 或 2048 bits，DOCSIS 3.1 則是至少要 2048 bits。

Table 5: DOCSIS 3.1 CM Code File

PKCS#7 Field	Description
signedData {	
Version	Version = 1
digestAlgorithmIdentifiers	SHA-1,SHA-256
contentInfo	
contentType	data (SignedContent is concatenated at the end of the [PKCS#7] structure)
certificates {	DOCSIS Code Verification Certification (CVC)
mfgCVC()	Required for all code files
mfgCVCCA()	Required for all code files using the new PKI
cosignerCVC()	Optional; required for cable operator co-signatures
cosignerCVCCA()	Optional; required for cable operator co-signatures using the new PKI when cosignerCVCCA certificate is not identical to the mfgCVCCA certificate.
} end certificates	
SignerInfo{	
MfgSignerInfo {	Required for all code files
Version	Version = 1
issuerAndSerialNumber	from the signer's certificate
issuerName	distinguished name of the certificate issuer
certificateSerialNumber	from CVC; Integer, size (1..20) octets
digestAlgorithm	SHA-1, SHA-256
authenticatedAttributes	
contentType	data; contentType of signedContent
signingTime	UTCTime (GMT), YYMMDDhhmmssZ
messageDigest	digest of the content as defined in [PKCS#7]
digestEncryptionAlgorithm	rsaEncryption
encryptedDigest	
} end mfg signer info	
MsoSignerInfo{	OPTIONAL; required for cable operator co-signatures
MfgSignerInfo {	Required for all code files
Version	Version = 1
issuerAndSerialNumber	from the signer's certificate
issuerName	distinguished name of the certificate issuer
certificateSerialNumber	from CVC; Integer, size (1..20) octets
digestAlgorithm	SHA-1, SHA-256
authenticatedAttributes	
contentType	data; contentType of signedContent
signingTime	UTCTime (GMT), YYMMDDhhmmssZ
messageDigest	digest of the content as defined in [PKCS#7]
digestEncryptionAlgorithm	rsaEncryption
encryptedDigest	
} end mso signer info	
} end signer info	
} end signed data	

4.3.2 Signed Content

The SignedContent field of the code file contains the CodeImage and the DownloadParameters fields, which may contain up to three items: a Root CA Certificate, a legacy Manufacturer CA certificate, and a Device CA Certificate. The final code image is in a binary format compatible with the destination CM. In support of the [PKCS#7] signature requirements, the code content is encoded as an OCTET STRING. Each manufacturer should build their code with additional mechanisms to verify that an upgrade code image is compatible with the destination CM

5 Generate Test CVC

製作 certificate 主要可以分成三步驟, 產生金鑰, 製作 CSR, 以及簽發憑證, 如果是要真的 certificate, 只要做完前兩步然後把 CSR 送給 CA, 簽約付錢就可以拿到 certificate 了, 如果只是要測試可以自己當 CA 簽發 certificate

雖然可以自己從 root certificate 開始一層層做下來, 但是 CableLabs 有提供 Test certificate 可以下載, 所以只要作 CVC 就可以了

5.1 Pre requirement

只要有 OpenSSL 和 Test certificate 就可以了

- OpenSSL
  使用 Linux 的應該都有了, 若是用 windows 的可以到 slproweb.com 或是 indy.fulgan.com 下載編譯好的執行檔, indy.fulgan.com 的版本相依的 DLL 比較少, 詳細的內容可以看 OpenSSL wiki
- Test Certificate
  CableLabs 提供了 Legacy PKI 和 New PKI 的 test certificate
  http://cablelabs.com/resources/digital-certificate-issuance-service/

5.2 Config Setting

Cablelabs 給的 test certificate 裡有詳細的步驟, 照著作就可以了

5.2.1 Legacy-PKI

先建立以下兩個檔案, 把 C 和 O 改成需要的值

csr_config.txt

```
[req]
default_keyfile     = TEST_DOCSIS_MFR_CVC_PRIVATEKEY.pem
default_md          = sha1
prompt              = no
distinguished_name  = req_DN

# Certificate Distinguished Name
[req_DN]
C   = <2 Digit Country Code>
O   = <Company Name>
OU  = DOCSIS
CN  = TEST Code Verification Certificate
```

ext.txt

```
extendedKeyUsage=critical,codeSigning
```

再用以下指令製作 RSA key 和 certificate

```
openssl req -newkey rsa:2048 -config csr_config.txt -out csr.pem -nodes
```

這行指令產生 2048 bits 的金鑰, 順便連 CSR 也一起做出來

```
openssl x509 -req -days 3653 -in csr.pem \
  -CA TEST_DOCSIS_CABLE_MODEM_ROOT_CA.PEM \
  -CAkey TEST_DOCSIS_CABLE_MODEM_ROOT_CA_PrivateKey.PEM \
  -CAcreateserial -extfile ext.txt -sha1 \
  -out TEST_DOCSIS_MFR_CVC.crt
```

這行指令用 CA 和 CAkey 簽發 TEST_DOCSIS_MFR_CVC.crt
接著就可以用 TEST_DOCSIS_MFR_CVC_PRIVATEKEY.pem 和 TEST_DOCSIS_MFR_CVC.crt sign image 了

5.2.2 New-PKI

DOCSIS 3.1 CVC 作法也是一樣的步驟

csr_config.txt

```
[req]
default_keyfile    = TEST_DOCSIS31_MFR_CVC_PRIVATEKEY.PEM
default_md         = sha256
prompt             = no
distinguished_name = req_DN

# Certificate Distinguished Name
[req_DN]
C  = <2 Digit Country Code>
O  = <Company Name>
CN = TEST Code Verification Certificate
```

ext.txt

```
extendedKeyUsage=critical,codeSigning
authorityKeyIdentifier=keyid,issuer
```

```
openssl req -newkey rsa:2048 -config csr_config.txt -out csr.pem -nodes
```

```
openssl x509 -req -days 3653 -in csr.pem \
  -CA TEST_CABLELABS_CVC_CERTIFICATION_AUTHORITY_PEM.CRT \
  -CAkey TEST_CABLELABS_CVC_CERTIFICATION_AUTHORITY_PRIVATEKEY.PEM \
  -CAcreateserial -extfile ext.txt -sha256
  -out TEST_DOCSIS31_MFR_CVC_PEM.CRT
```

5.2.3 ca comomand

以上的作法是用 x509 指令, 這個指令不能設定很精確的時間, 可以使用 ca 指令來達成

ca.conf

```
unique_subject = no

[CA_Legacy]
dir            = ./Legacy-PKI
database       = $dir/index.txt
new_certs_dir  = $dir/newcerts

certificate    = $dir/TEST_DOCSIS_CABLE_MODEM_ROOT_CA.PEM
serial         = $dir/serial
private_key    = $dir/private/TEST_DOCSIS_CABLE_MODEM_ROOT_CA_PRIVATEKEY.PEM

default_startdate = 100101000000Z
default_enddate = 200101000000Z
default_md     = sha1

policy         = policy_Legacy
copy_extensions = none
x509_extensions = ext_Legacy

[CA_New]
```

```
dir            = ./New-PKI
database       = $dir/index.txt
new_certs_dir  = $dir/newcerts

certificate   = $dir/TEST_CABLELABS_CVC_CERTIFICATION_AUTHORITY_PEM.CRT
serial        = $dir/serial
private_key   = $dir/private/TEST_CABLELABS_CVC_CERTIFICATION_AUTHORITY_PRIVATEKEY.PEM

default_startdate = 100101000000Z
default_enddate = 200101000000Z
default_md     = sha256

policy          = policy_New
copy_extensions = none
x509_extensions = ext_New

[policy_Legacy]
countryName            = supplied
organizationName       = supplied
organizationalUnitName = supplied
commonName             = supplied

[policy_New]
countryName            = supplied
organizationName       = supplied
commonName             = supplied

[ext_Legacy]
extendedKeyUsage=critical,codeSigning

[ext_New]
extendedKeyUsage=critical,codeSigning
authorityKeyIdentifier=keyid,issuer
```

使用 ca 指令需要把一些資料夾和檔案先設定好, openssl 不會自動產生這個檔案，以這個設定檔為例，要在執行 openssl 的路徑下建立 Legacy-PKI 和 New-PKI 兩組檔案，把 CableLabs 的 test certificate 和 key 放到相對應的位置，newcerts.txt 是 certificate 存放的路徑，index.txt 是 database，建立一個空檔案即可，serial 是序號，每次簽發都會自動加 1，初次使用要給初始值，例如：01。

- unique_subject
  這項設定用來查 subject 是否重複，因為測試時可能會同一個 certificate 重複簽發，關掉會比較方便
- default_startdate default_enddate
  預設的 certificate 有效日期，對應的指令為 -startdate -enddate
- default_md
  預設的 message digest，Legacy-PKI 用 sha1 New-PKI 用 sha256
- policy
  policy 設定中標記為 supplied 表示一定要有這一項，沒出現的欄位會被移除

全部設定好後要作 CVC 就很簡單了, 先用之前的方式把 CSR 做好，再用以下指令製作 certificate。

```
#DOCSIS 3.0 CVC
openssl ca -config ca.conf -in csr.pem -name CA_Legacy -batch

#DOCSIS 3.1 CVC
openssl ca -config ca.conf -in csr.pem -name CA_New -batch
```

```
#要改時間也很方便
openssl ca -config ca.conf -in csr.pem -name CA_New -batch \
  -startdate 150102030405Z -enddate 250102030405
```

5.3 Sign image

把 cvc 和 key 做好後就可以拿來 sign image 了

1. 製作 code file

   根據 code file 的格式把 image 和 DownloadParameters 做成 SignedContent，最常見的情況是 DownloadParameters 裡是空的，這種情況只要在 image 前補上 0x1C 0x00 0x00 就可以了。

2. Sign image

```
# DOCSIS 3.0 用 sha1
openssl smime -sign -nosmimecap -outform der -md sha1 \
  -binary -in codefile -out pkcs7 \
  -inkey TEST_DOCSIS_MFR_CVC_PRIVATEKEY.pem \
  -signer TEST_DOCSIS_MFR_CVC.crt

# DOCSIS 3.1 用 sha256 並加入一張 CVC CA cert
openssl smime -sign -nosmimecap -outform der -md sha256 \
  -binary -in codefile -out pkcs7 \
  -inkey TEST_DOCSIS31_MFR_CVC_PRIVATEKEY.PEM \
  -certfile  TEST_CABLELABS_CVC_CERTIFICATION_AUTHORITY_PEM.CRT \
  -signer TEST_DOCSIS31_MFR_CVC_PEM.CRT
```

   OpenSSL release 的版本不能指定 sign time，要設定 sign time 可以改系統時或是用改過得版本。[2]

3. 合併

   smime 做來的檔案只有 PKCS7 ，做成 code file 只要把 PKCS7 和 SignedContent 併在一起即可

[2] https://github.com/jianiau/openssl

A CableLabs Test Certificates

A.1 Legacy-PKI

A.1.1 TEST_DOCSIS_CABLE_MODEM_ROOT_CA_PRIVATEKEY.PEM

```
-----BEGIN CERTIFICATE-----
MIID2DCCAsCgAwIBAgIIEBAQEBAQEBAwDQYJKoZIhvcNAQEFBQAwgZcxCzAJBgNV
BAYTAlVTMTkwNwYDVQQKEzBEYXRhIE92ZXIgQ2FibGUgU2VydmljZSBJbnRlcmZh
Y2UgU3BlY2lmaWNhdGlvbnMxFTATBgNVBAsTDENhYmxlIE1vZGVtczE2MDQGA1UE
AxMtRE9DU0lTIENhYmxlIE1vZGVtIFJvb3QgQ2VydGlmaWNhdGUgQXV0aG9yaXR5
MB4XDTAxMDIwMTA3MDAwMFoXDTMxMDIwMTA2NTk1OVowgZcxCzAJBgNVBAYTAlVT
MTkwNwYDVQQKEzBEYXRhIE92ZXIgQ2FibGUgU2VydmljZSBJbnRlcmZhY2UgU3Bl
Y2lmaWNhdGlvbnMxFTATBgNVBAsTDENhYmxlIE1vZGVtczE2MDQGA1UEAxMtRE9D
U0lTIENhYmxlIE1vZGVtIFJvb3QgQ2VydGlmaWNhdGUgQXV0aG9yaXR5MIIBIjAN
BgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAwxA1cK01p2CopSnHES8r/BQyObM1
DWQvrs77VlOftuqx0iJPXBJ86gqyfQoO+5Od5c07GFxhN1Ca7GA98AdBA9vsNBBC
fP19bQhiHEO3sHKhyMLLYNWsmWEFPT0vcOrBd3fdjsi2D7fRtMWj2gK7nF84k7mg
tTqVNGZ5JBbqGVnV108Gszkype/Y/8mS+YJMk61SRd5pksqG9baLmH7BMiRXWp1F
HHzlsN0ityXLJEXvNBSlP6HkHsM9ntDkSEuqYmnaef50x3LrPpHWdzXgS1WkAyfv
wPGbaJJH+a+uLlEmdiCqmDPSNkxGt4wh6rEtGBTU/HLDviIvMhOd3KNODwIDAQAB
oyYwJDASBgNVHRMBAf8ECDAGAQH/AgEBMA4GA1UdDwEB/wQEAwIBBjANBgkqhkiG
9w0BAQUFAAOCAQEAwPdm9moSKzQdxCsH5UO6FpYDAJlRItO+jyt/+8FG7bxfdXXB
sLYa7v5QOUluczV9k8aBy0UJ406WT3SXnDIMFIBJimZ02mMNMAnHqEVgkrjDTxKz
jQ2cQmvZMT/XYaG0zLuK7mUI+awg5/L2+KwWF3ONI5gCgxzodCQLnwdzFq/g/WYH
0dayTu/PkAgcWSy3ight6z4a8efWkLhlWytAwnxxliB7UcaKlt12qSp/GkEGb9zw
uG7Mw87myDVA3VaZ4SqcUFVO0F5+aWwb/MtkqJjUBS9KBwR/7tLx9XPcNEY/kbTH
QE2q1ah4IOstdJeXNa81HA8JtM8ElK/SdymZXQ==
-----END CERTIFICATE-----
```

A.1.2 TEST_DOCSIS_CABLE_MODEM_ROOT_CA_PRIVATEKEY.PEM

```
-----BEGIN PRIVATE KEY-----
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQDDEDVwrTWnYKil
KccRLyv8FDI5szUNZC+uzvtWU5+26rHSIk9cEnzqCrJ9Cg77k53lzTsYXGE3UJrs
YD3wB0ED2+w0EEJ8/X1tCGIcQ7ewcqHIwstg1ayZYQU9PS9w6sF3d92OyLYPt9G0
xaPaArucXziTuaC1OpU0ZnkkFuoZWdXXTwazOTKl79j/yZL5gkyTrVJF3mmSyob1
touYfsEyJFdanUUcfOWw3SK3JcskRe80FKU/oeQewz2e0ORIS6piadp5/nTHcus+
kdZ3NeBLVaQDJ+/A8Ztokkf5r64uUSZ2IKqYM9I2TEa3jCHqsS0YFNT8csO+Ii8y
E53co04PAgMBAAECggEAEn7I8mQaAELT+sUZUPHOE9q+YGYzk496TfLmNes765YZ
QjlqmOpP2ckl32FCxmvavt1UyxyHQz1rUxpVBRtK+NizbsRT6ZIHgH0ZAGETW7Gi
0WxW6FmaQfjHDGVuyCAaM/+zjERPrvHY+nD69kLoEw2cSbWCHWzulMdamSdrDvgg
5EKKxLZwU5LXhfzZAJKehCBCEXaNxmuVKYk4tijzTw2WFzHbpKztRxl1F3VLVDRX
RdeD2YWHiBvKzM7Mrw0kCGGSAfTPeLAuiq086DxGApiYsl8FFQZAyOavDphAWBYo
17+PtC8tGCeSdvHeUvs3P8a3PK/2r1aFQ8Tt+8IikQKBgQD/MqSaSkB2Yb461CfE
rBeI2tQ1U7R4Xz8zI+AME4SuNl9szhNoH0eoso2GOCMlq2kYVOPTKV81l3k8OGkx
ZZswqrDWUnHxVCTmTmtiUq7b3C2/eawvt+KE/5efKR1xQ1bU8znsE9u7PFZ+IWbK
Uh1AOSovTUMFlVcI5WE22pd6/QKBgQDDrSz5/qIDdG3NmHXEMRbCdJfuUnUeEPcI
y2UyAvjcMOB4pcDeSRHklTpDY83hpA0PzIl9bMzlgCYLQz00QEv95fZiWdu338AV
yyoDh+3V7YYuZ7aaNuXQ0k0jXX6W4PzAgCppqTh63cDxbtR19vBHtgzhsTJnwZX6
8rbHi84Y+wKBgQCM+wA+EFkOTS8XNs8FcoDJ5QIot6ZSfWPfr6j9YucAix8qb6n+
8pDW3FUc2fRowgocHAGETS3A4H5kS1GprVUPjKyGqiMyS+baqGXgeocJBNjtF52M
+wwTp1Ou5LrUMHxOxlOwXMQd6tZpdbBggv6P1US9vvQQqJDdZELRV+8pTQKBgHMo
```

```
MoT5tw1Dcu+BHyWUIinUhilqe6RzJX8WfHqfRygG18QJAGWRKSeWLxDD5sE5U7qN
jAi1hCw4a6tLKfTbNh25PXQKIAWpd5kb1KD5WR4CSGp8/Pjq//sSGPWMK1jOFdGQ
W+WNAkFEyi0MJZs69ZOubwxPxTNMuWBTgQAUpfkLAoGAatLdk/lwNzAvT5LBhf98
AmI7ErzOE9RU2qj8PfU+o+XhusVlDAqJwH4HXjwCd7po1sD99D+MjWpqzCdWO+ka
6K45DgZbxJ/sLK35+i7+sDSg00eNSeZfs5BYGXRKQThy0GT83zAkp+uLb0Spt/jE
eyr7xOl2JDZ9A5USc4TLqNI=
-----END PRIVATE KEY-----
```

A.2 New-PKI

A.2.1 TEST_DOCSIS_CABLE_MODEM_ROOT_CA_PRIVATEKEY.PEM

```
-----BEGIN CERTIFICATE-----
MIIFSTCCAzGgAwIBAgIJAOjhI1VGwwEdMA0GCSqGSIb3DQEBCwUAMHAxCzAJBgNV
BAYTAlVTMRIwEAYDVQQKEwlDYWJsZUxhYnMxFzAVBgNVBAsTDlRFU1QgUm9vdCBD
QTAxMTQwMgYDVQQDEytURVNUIENhYmxlTGFicyBSb290IENlcnRpZmljYXRpb24g
QXV0aG9yaXR5MCAXDTE1MDEyMjIzNTI0MVoYDzIwNTAwMTIyMjM1MjQxWjBuMQsw
CQYDVQQGEwJVUzESMBAGA1UEChMJQ2FibGVMYWJzMRYwFAYDVQQLEw1URVNUIENW
QyBDQTAxMTMwMQYDVQQDEypURVNUIENhYmxlTGFicyBDVkMgQ2VydGlmaWNhdGlv
biBBdXRob3JpdHkwggGiMA0GCSqGSIb3DQEBAQUAA4IBjwAwggGKAoIBgQCtpssp
fXi9x1eBgE3SnB5yMq486yhx6ZilCJJ5FPINcdJGKa7aW0qxQlvljyjClTOmw18a
3uXB/F2bWBBkTBg4HxdVVNQHuwlihLZfz+bTG2CN6ZP6ByyM81iW8DfXn1fEoIKG
/H5WJSYfFd8ZPQHOzqVrnc3/Dpe9L5bQHlpvwWNT+xuJZ7u3iMyeoHiq6/wVLXk7
7qL/ISso8DeDqLB+p3ZV0dfFwUUQEnmGDAx4X/LyIGRSScfYx9j0zkSVelGRRgoa
oTTbffsqJfJdtHTlrOfsBSsr6gUGifDGFXFC4dXJV/ldLJmm+48H6XwDYkm182m9
Joo+2QM6v7sjVKwBwuFJ/1uSFHIM0B5laZj1C8KUcfSUqBqM1bsxtOxmLdqv26B9
bsuvFfoOv+W6o3BrolGo4CYNK87Ic6+vML0UnQ7QUv12hdNZ0AZG3Ac/OrCUIvS2
EWkQAt+gUJTb9CtBgyVJ+flS7e2ItrbTaAOB54OvY4f+3k6GrKsSuY6WBXcCAwEA
AaNmMGQwDgYDVR0PAQH/BAQDAgEGMBIGA1UdEwEB/wQIMAYBAf8CAQAwHQYDVR0O
BBYEFIBYNvRodhQYM5UbdnmLqSh82KaiMB8GA1UdIwQYMBaAFIlieT20B8nzxpdZ
3bbcZQszVP/7MA0GCSqGSIb3DQEBCwUAA4ICAQCvsFYNgL24bm53P5uYfE63lokw
xy/GCas2kYF8AlFErmPG9w3P+uDorpjGDxZpFE7H8JHrOlpRJ0b0iexBDzEWeAHN
73k4hEch2WUvS1ZC0EK+NA+/TKWIAWkl9TDDLbVxuF6SjZYvMMoTJI/8lK7Hrutm
HumLVXHKevlidaa+XQA/p3OnG9xYHJbIvl0XyMENp+rHhstoDBpV1VzIoKn9pquF
+4Nl/fxr548PusWsTgrFIDyveurdBn2dEyieilHyTm1Tn8d/MWqTYFimSznokvVX
8HOJDLOXi1jngfhDfj5s4WoNvu+NCwsg1mym1AQ4OeOEXTcAb4DbibNDbp3iAJcj
PKkq1xsDU1n+lGzHebWosh2yXtzccc2l++/90VDsfZeuOGebec7KxRxIWEa4ea13
tcsKF0eZ0soEeK1IezjXHxgO/dKWpkbOsUGRjLtqINgtTnKIsA5w6V+M6LErs8xO
RcjN17zVACNrQWallk/P2z19Bp6Gv3TBcFlSwrHTJDNk59xYLy9V2HLpaUCs7guS
h4/ndoLwI1dIKnBpYHUhIrXgTRSxQicOKFzXbvK5vbBHp57Mh8a+7bfylZC/7xVt
U29DVRsUYuAC3zsYhRFeQiSBoUB2aQQz11t7gwaBhPjall4dFqGiNDSFzNLavK93
3frISqGMdFXD9etuHg==
-----END CERTIFICATE-----
```

A.2.2 TEST_CABLELABS_CVC_CERTIFICATION_AUTHORITY_PRIVATEKEY.PEM

```
-----BEGIN PRIVATE KEY-----
MIIG/gIBADANBgkqhkiG9w0BAQEFAASCBugwggbkAgEAAoIBgQCtpsspfXi9x1eB
gE3SnB5yMq486yhx6ZilCJJ5FPINcdJGKa7aW0qxQlvljyjClTOmw18a3uXB/F2b
WBBkTBg4HxdVVNQHuwlihLZfz+bTG2CN6ZP6ByyM81iW8DfXn1fEoIKG/H5WJSYf
Fd8ZPQHOzqVrnc3/Dpe9L5bQHlpvwWNT+xuJZ7u3iMyeoHiq6/wVLXk77qL/ISso
8DeDqLB+p3ZV0dfFwUUQEnmGDAx4X/LyIGRSScfYx9j0zkSVelGRRgoaoTTbffsq
```

```
JfJdtHTlrOfsBSsr6gUGifDGFXFC4dXJV/ldLJmm+48H6XwDYkm182m9Joo+2QM6
v7sjVKwBwuFJ/1uSFHIM0B5laZj1C8KUcfSUqBqM1bsxtOxmLdqv26B9bsuvFfoO
v+W6o3BrolGo4CYNK87Ic6+vML0UnQ7QUv12hdNZ0AZG3Ac/OrCUIvS2EWkQAt+g
UJTb9CtBgyVJ+flS7e2ItrbTaAOB54OvY4f+3k6GrKsSuY6WBXcCAwEAAQKCAYAq
jm7JmztE3x592TC5RZNcjzkOKt89k27aOxCSZeSwEM3kbgCw+KCEcmvNBVmK3j/L
RaQAFNIfyTYzhNB3lAJ3dn8kZWaVAmxZXYbIUk4SPGpcPPsL0khDvrteOj6lNCKx
SFjUtfpMKEyAX77OE5pm0VPO3NyH/k6HiasfE+E79Bvgj/B+qxJg90CWO2N3F+Au
XHeSjXgaqElIfh5MwMGIOIgSnhuQFj5TowNvK3K+6VzDo9aSKAy2/rsXu5kpPNFq
zziKUrKjolzIoLSubfxOr+0nq5PFmViiVDfCdYise4Bjx4cFEouzjA00MhimpayH
8P0HZEEr3MLmGmWi0Ga1GXN1h2QdauZRMfVBnk22DOXXYsHmP3xianVMJq3Kih7/
jJmjos2q1tvfPjpLDL2A0AWmxUDgA1hGmQFbcpS6Z5Ogkbsh1VyjC7x07r5PZ6J2
1zMFQc/tBED7O4L8zDfnHMSYvxKyUAndEVTK8gTvNTXm1VOglIxVdUeG3/EGSnkC
gcEA5wjycuZkXc0ZSO+5Egxyd88KdmielhCLfgKXWoCzYsgZ4Cn81t7PPZgL7t8w
KUWnqhisLxPAOwRShbKUytUwuQ0Tf4KFtqxJa97I3XKs/fevfcTqMVAnj/Qpqonw
MwjvTy49R+7EjWmNWzgE6JxaSeOPPz9mIWmZ3g4+lTeo+wRXHSo9jNHP8IgkSECV
rkomrs47+2tfxaq1ZKJ3FbEwumK9YJ411de5jw7bG0fjBzoZ0OIQp4Snx39G8kRr
reONAoHBAMBqd0ACZep9+1GjrHA558zH835uIIfOH0525gqR91YIOAZyBK0LCnAm
oNIO23wIh+FWFCWSPwXjV5vtKfXltR6YoEz8CT+Jca06O6L4/RQHwVkBZwLHDS1z
UkV7kyghsO8JdylAcQFvnvzp1QlypsmXw8E7QRqqDd2LdSSOvr7hq8JPyiMLbrk9
8dPWzCV+ftu5lldFJ5wLyMHI9FftT136C3jE3B/g2gMzP8PwgJrnenomEDMuaNrK
TBJwg/0qEwKBwQCOUXmghAIhsj04K8vyrU46NYT5afr+8BQtvITl0FAwspkV+Gdz
KW+6PpZjQINXeTwOUQKIQX9yG3iGOXrk86z+4Wo4avvZionz9BoCKCkejVxCenHV
jM3CJKMiFbSsi313ZkGDtTbI8dOM9JY5gE8yIwbT7EcKoBWTr2yn2NC0SHktUx71
Ry3zurm29GoAIldv6UUT3JHznvP8mQQIjc05k6ebA/qfzzLUSE96n67ffc/3V7rw
powAHYxjovg99uUCgcEAlrmPakO2pAZFvQDSTiL98grbMTmBIACVw3gd4T/QGMNd
Z6cfBJI2ff20Uney9KkWD6zHIVs7JRieAxW3ndhvlWUHhOaVNAEtW/3Ww2X5kRw2
F8uibqmQJ/9C/gy8DF6/469teZHyM6bFSua8q6b1ActxW0dYS1PUgqwAej+bKZlL
W233MJZ/CHp+mWuUBPbe1q29F+WjDniorSwGQ24wkrQREmxa5lnTTVhy5cDabP1Q
kqiIXh9HocN/7Z3XtpOjAoHAStyVd3gzHcFPHAE22ySjK6IHKVUR2A7NzJN145g4
5TFj2q85Cn4syN/VIAHOnFvslrLbvzeEiLUF5DTBg0vKReH22obIEhIvoqb18DRy
WUsGV5LP8fr9xZtuop12Do6u0T+r+DbPnmORlCCNLxaTgKAwWBSI7T9ndeQd/vEs
B6ENl5d8GVPuaeUSOBkIsmPVNjBqT40wNH2NXUEIWXS6VyC2FopHXlAPEStNwD8Y
jCzDBo2spJVfhdtL0nWBvo3m
-----END PRIVATE KEY-----
```