TWISC@NCTU

交大資通安全研究與教學中心

DNSSEC
Authoritative Server
建置 SOP 簡易版

版本： 2.12

TWISC@NCTU 主任：謝續平教授

參與人員：李冠毅、陳柏愷

本研究成果由教育部補助

目錄

1. 前言

在此份 SOP 中，我們將以 BIND 架設支援 DNSSEC 之 authoritative server，並將建置及維護之流程做一完整的說明。

此版本著重於較精簡的作法，減輕佈署與維運的負擔，強調高可用性。跟完整版的差別主要在於，此方法佈署後不須做定期的金鑰維護；不使用 nsupdate，修改 zone file 的方式與未導入 DNSSEC 前相近；不介紹 DLV 僅使用 DS。

特別要說明的是金鑰管理的部分，在最高安全性的考量之下，會使用 ZSK 與 KSK，並設定金鑰使用期限，但這樣的方式在管理上是比較高的負擔，故本版本不採取此作法。本版本以管理簡便為考量，僅採用沒有期限的 2048 bits KSK，並且不使用 ZSK，所以不須做金鑰管理。

而在安全的強度方面，根據”European Network of Excellence in Cryptology II”於 2010 年 3 月的報告指出，破解 RSA 2048bits 約需要三億美金的預算，使用 ASIC 花費 10 年來運算。也就是說，本文所建議的方式，仍然是需要非常高的代價才可被破解，而破解後的收穫僅是獲得 DNSSEC 的金鑰，可用來仿冒 Resource Record。我們認為，除非管理者所管理的相關系統，價值超過數億美金， 否則本文所建議的方式已經提供足夠的安全性。

若讀者對於 DNSSEC authoritative server 其他可行的做法有興趣，可另外參考完整版。

2. 系統安裝程序

2.1. 在 Ubuntu 中以 aptitude 安裝 BIND

作業系統版本：Ubuntu 13.04

以管理者身分執行如下指令

```
root@Ubuntu:~$ aptitude install bind9 dnsutils
```

2.2. 在 CentOS 中以 yum 安裝 BIND

作業系統版本：CentOS 6.2

以管理者身分執行如下指令

```
root@CentOS:~$ yum install bind
```

設定使用 rndc 時所需的 rndc.key

```
root@CentOS:~$ rndc-confgen -a -r /dev/urandom
root@CentOS:~$ chown root:named /etc/rndc.key
root@CentOS:~$ chmod 640 /etc/rndc.key
```

注意：原則上此份 SOP 以 Ubuntu 為主，CentOS 部分為補充資料。

2.3. 在 FreeBSD 中以 Ports Collection 安裝 BIND

作業系統版本：FreeBSD 9.0-RELEASE

BIND 9.8 系列位於 ports tree 中的 dns/bind98 下，我們先將目錄切換至 BIND 9.8 系列所在的位置：

```
# cd /usr/ports/dns/bind98
```

接著進行編譯前的設定：

```
# make config
```

這裡必須選取其：

- SSL：提供 DNSSEC 所必須之簽章簽署及驗證功能。
- REPLACE_BASE：取代 base system 中之 BIND（在 FreeBSD 8.2 中是 BIND 9.6.x），使系統啟動服務時採用新版。
- SIGCHASE：使 dig 擁有 DNSSEC 的驗證功能，非必要功能但可協助除錯。

最後進行編譯及安裝

```
# make install clean
```

注意：原則上此份 SOP 以 Ubuntu 為主，FreeBSD 部分為補充資料。

3. 系統初始設定

3.1. 將 BIND 設定為開機時自動啟動

作業系統版本：Ubuntu 13.04

以 aptitude 安裝完成後 BIND 即會於開機時自動啟動。

作業系統版本：CentOS 6.2

以管理者身分執行以下指令

```
# chkconfig named on
```

BIND 即會於開機時自動啟動

作業系統版本：FreeBSD 9.0-RELEASE

以編輯器開啟 /etc/rc.conf 設定檔：

```
# vi /etc/rc.conf
```

並於設定檔中加入

```
named_enable="YES"
```

BIND 即會於開機時自動啟動

3.2. 設定 named.conf.options

開啟 named.conf.options

```
root@Ubuntu:~$ vi /etc/bind/named.conf.options
```

將檔案內容調整如下：

```
options {
   directory "/var/cache/bind";

   dnssec-enable yes;
   dnssec-validation auto;
   dnssec-lookaside auto;

   // Authoritative-only Name Server
   allow-query-cache { none; };
   allow-query { any; };
   recursion no;

   // Set Secure Default
   allow-transfer { none; };
   notify yes;

   auth-nxdomain no;    # conform to RFC1035
   listen-on-v6 { any; };
};
```

於工作目錄產生附屬設定檔 named.conf.local，將所屬的 zone 設定放入此檔案中。

```
root@Ubuntu:~$ vi /etc/bind/named.conf.local
```

以 example.com 為例，在 master 上此設定檔內容如下：

```
zone "example.com." {
   type master;
   auto-dnssec maintain;
   update-policy local;
   allow-transfer { slaves_list };
   file "/etc/bind/example.com/db.example.com.signed";
   key-directory "/etc/bind/example.com";
};
```

注意：請自行將 slaves_list 更換為所架設之 slave server 列表！

其中 auto-dnssec 最為重要，必須將其設為 maintain，BIND 才能夠自動進行金鑰的更換及網域的簽署，而且需搭配設定 update-policy local 參數，否則 named 會無法啟動或 auto-dnssec 會無更新權限使 DNSSEC 因 RRSIG 過期而無法運作。file 及 key-directory 則分別為 zone file 及 DNSSEC 所需金鑰之存放路徑。

這邊特別要注意的是，我們所引用的 zone file 為 db.example.com.signed 而非 db.example.com，後續管理員應手動修改 db.example.com，然後簽署成 db.example.com.signed 讓 BIND 載入。

若您現在設定的為 slave，則此設定檔內容如下：

```
zone "example.com." {
   type slave;
   masters { masters_list };
   allow-transfer { none; };
   file "/etc/bind/example.com/db.example.com.signed";
};
```

注意：請自行將 masters_list 更換為所架設之 master server 列表！

生成 zone file 及金鑰存放的資料夾：

```
root@Ubuntu:~$ mkdir /etc/bind/example.com
```

在 master 上生成 zone file：

再此強調，此 zone file 並非是 BIND 最後導入的 zone file，BIND 應導入有簽章的 zone file

```
root@Ubuntu:~$ vi
/etc/bind/example.com/db.example.com
```

db.example.com 範例：

```
$TTL    600
@   IN  SOA example.com. admin.example.com. (
          1      ; Serial
          3600    ; Refresh
          600     ; Retry
          86400   ; Expire
          600     ; Negative Cache TTL
          )
;
@   IN  NS ns.example.com.
@   IN  NSEC3PARAM 1 0 100 61
ns  IN  A 127.0.0.1
```

其中 NSEC3PARAM 是用來指示 BIND 使用 NSEC3，並提供使用 NSEC3 時的各項參數。

在此例我們採用 Iteration 的值為 100，根據我們的安全分析，這樣的值才具有足夠的強度以抵抗暴力猜測法攻擊。而使用 salt 同樣能增加安全強度，此例的 salt 為 61，使用者可採用任意的十六進位字串 (0-F 且長度為 2 的倍數) 增加亂度，以抵抗比對法攻擊。

3.3. 金鑰生成

這個章節會介紹如何生成簽署網域所需要的金鑰，由於管理金鑰及簽署網域皆由 master 完成，建置 slave 時可跳過此部分。

以下為金鑰生成指令範例：

```
root@Ubuntu:~$ dnssec-keygen \
-a NSEC3RSASHA1 \
-b 2048 \
-f KSK \
-r /dev/urandom \
-K /etc/bind/example.com \
```

```
example.com
```

參數說明：

1) –a：選擇金鑰演算法
2) –b：設定金鑰長度
3) –f：金鑰 flag 設定
4) –r：亂數來源
5) –K：產生後之金鑰存放目錄
6) Zone name

此指令會生成一對沒有使用期限的金鑰，放在兩個開頭為 Knnnn.+aaa+iiiii 的檔案，同樣放置於該網域的 key-directory。需要先將生成的檔名紀錄下來，後續的建立信任鏈的過程將會用到這個資訊。

3.4. 簽署網域

簽署 zone 的指令如下：

```
root@Ubuntu:~$ dnssec-signzone \
-3 61 \
-H 100 \
-K /etc/bind/example.com \
-o example.com \
-S \
-u \
-z \
/etc/bind/example.com/db.example.com
```

參數說明：

1) -3：NSEC3 使用的 salt 值
2) -H：NSEC3 使用的 Iteration 值
3) -K：存放金鑰的資料夾
4) -o：網域名稱
5) -S：Smart signing

6) -u：更新 NSEC/NSEC3
7) -z：使用 KSK 來簽署整個 zone file
8) zone file

這個指令會於存放金鑰的資料夾中挑選出適當的金鑰來簽署整個 zone file，並產生一個檔名為“原始檔名.signed”的已簽署 zone file，放在與原本 zone file 同一個目錄，這個簽署過的 zone file 才是 BIND 真正載入的 zone file。

3.5. 權限設定及結果驗證

網域簽署完後，需要讓上述設定中新創檔案的擁有者與 named 執行者相同。根據採用的作業系統與 BIND 安裝方式的不同，named 的執行者可能不同，舉例來說，若前述產生金鑰的階段是使用 root 身分產生，但是 named 是由 bind 這個使用者身分來執行則會產生錯誤。在重新開啟 named 服務前，可執行如下指令更改網域目錄擁有者：

```
root@Ubuntu:~$ chown -R bind /etc/bind/example.com
```

經過上述的設定，此時我們可以如下指令指示 BIND 重載其設定檔：

```
root@Ubuntu:~$ rndc reload
```

設定檔重新載入後，我們可以用 dig 來驗證設定是否生效：

```
root@Ubuntu:~$ dig +dnssec +multiline -t dnskey
example.com. @127.0.0.1
```

若回應中包含如下的 DNSKEY 及 RRSIG 則表示設定生效

```
example.com. 5226 IN DNSKEY 256 3 8 (
AwEAAdNW7YIhcTdqXrzgZjJJ35VjAFT1ArvnhAzXDm7A
uGxSQqmGBRmjJvBv0xS4gahB9mj6ekF0dVKoeZgLmNAj
```

```
o8hj2JI7K281YTo2R5k3mKSc4hOCP55hR22r5hIsPJoT
19pv/VdZQfyTzZ96frQ16qRa9+/GSjzjtFfQv16FwE7R
) ; key id = 55231
example.com. 5226 IN DNSKEY 257 3 8 (
AwEAAagAIKlVZrpC6Ia7gEzahOR+9W29euxhJhVVLOyQ
bSEW0O8gcCjFFVQUTf6v58fLjwBd0YI0EzrAcQqBGCzh
/RStIoO8g0NfnfL2MTJRkxoXbfDaUeVPQuYEhg37NZWA
JQ9VnMVDxP/VHL496M/QZxkjf5/Efucp2gaDX6RS6CXp
oY68LsvPVjR0ZSwzz1apAzvN9dlzEheX7ICJBBtuA6G3
LQpzW5hOA2hzCTMjJPJ8LbqF6dsV6DoBQzgul0sGIcGO
Yl7OyQdXfZ57relSQageu+ipAdTTJ25AsRTAoub8ONGc
LmqrAmRLKBP1dfwhYB4N7knNnulqQxA+Uk1ihz0=
) ; key id = 19036
example.com. 5226 IN RRSIG DNSKEY 8 0 172800
20111025235959 (
20111011000000 19036 .
NM23qUtTFHno+V6TVGO5cyohgVZEBLWDMRnI8lv78SWs
YJFbBP0xWERfHloiWZhOoONsbXn2QAWbYaR0Iiysu64A
hree92lFGdXaFYrISRWQJo6LOixc7gvorrPwpeNG9euA
FZEO94Z7SetqGXo4uPwuxQLen14MLOBjE5Hjj+dyuBQk
yY4Hr2CElyuEbVR5EosxZhcziJFpJrO7LDSOxbyfw8V+
o2Lw9zu+nQSgIBOxlAgX0Jd/SyvcmmCTMuz8CsvSAVFY
nJ4G5mGK052dqMzFn9U38E+6fUh/UpPHHDRCUjFSwI/l
ZnfoebROk5+nuk0gJGUSrdPyoCfsiMNbKQ== )
```

3.6. 建立信任鏈

網域中的的紀錄要能夠通過驗證必須提交網域金鑰的 Hash 給上層網域，達成上層網域替下層背書的效果，在概念上類似 DNS 原有的 NS。

建立信任鏈的機制有 DS 及 DLV 兩種。DNSSEC 在設計上，最終希望以 DS 串成階層式信任鏈架構，然而在 DNSSEC 推廣的初期，因 DNSSEC 佈署不夠普及，因此仍需要一個中央控管型的信任鏈機制，DLV 即為此設計。

在本文中僅描述 DS 的做法。

3.6.1. DS

將目錄切換至放置金鑰的存放目錄下，輸入以下指令：

```
root@Ubuntu:/etc/bind/example.com$ dnssec-dsfromkey
Knnnn.+aaa+iiiii
```

Knnnn.+aaa+iiiii 為先前紀錄之金鑰檔名，將產出之資料提交給上層的網域。

舉例來說，產生的資料大約如下：

```
example.com. IN DS 15631 5 1
3E2C801514374F275A89480AA3997895EC6DBDF9
example.com. IN DS 15631 5 2
5619C75FCABBB34F633DAC3D08BDCEB29D46A3E36EE0E2CC7F
48726A 34C04A45
```

上層網域拿到 DS 紀錄後處理方式與設定 NS 類似。

匯入完成後，可在任何機器使用如下指令查詢是否匯入成功：

```
root@Ubuntu:~$ dig +dnssec -t ds example.com
```

3.7. 完整驗證

以上步驟若正確，應已完整佈署 DNSSEC authoritative server，可透過以下指令驗證

```
root@Ubuntu:~$ dig +dnssec -t soa example.com
@dnssec-resolver
```

在這裡應使用此 DNSSEC authoritative server 之外的 DNSSEC resolver 來解析，若回應中出現 ad　flag，則表示建置正確。

4. 系統維護程序

當 DNSSEC 伺服器架設完成後，若 RR 需要新增修改、DNSSEC 運作失常或遭到入侵時可參考此章節。

4.1. RR 增刪修改

若是管理者要直接修改網域的 zone file，因為涉及簽章的更新，需要有五個步驟

1) 修改原始 zone file
2) 凍結 zone
3) 產生已簽署 zone file
4) 修改已簽署 zone file 的 owner
5) 解凍 zone 使其生效

這邊要注意的是 BIND 設定裡真正使用的 zone file 為 db.example.com.signed，而使用者手動去修改的 zone file 為 db.example.com，然後再以指令重新簽署而產生新的 db.example.com.signed，並使其生效。

實作指令如下

1) 修改原始 zone file

我們先修改原始 zone file(db.example.com)，方法與傳統管理 DNS 時皆相同。

2) 凍結 zone

執行以下指令：

```
root@Ubuntu:~$ rndc freeze
```

這個指令會把目前的 zone data(db.example.com.signed)寫回，並凍結動態更新機制。在凍結的時間內，BIND 仍以當時的 zone data 正常提供服務，不會有服務中斷的情況，並且在解凍之後，會重新讀取 zone file(db.example.com.signed)來提供服務。

3) 產生已簽署 zone file

簽署 zone 的指令如下：

```
root@Ubuntu:~$ dnssec-signzone \
-3 61 \
-H 100 \
-K /etc/bind/example.com \
-o example.com \
-S \
-u \
-z \
/etc/bind/example.com/db.example.com
```

這個部分所做的事情就是由原始 zone file (db.example.com)生成簽署過的 zone file (db.example.com.signed)，而新產生的簽署過 zone file 會蓋掉舊的簽署過 zone file。作法與章節 3.4 相同。

4) 修改已簽署 zone file 的 owner

這個部分依使用的 OS 或管理者的作法可能有所不同，譬如說以 root 來執行前一步時，產生的新的簽署過的檔案，其 owner 會為 root，如果該 OS 是以 bind 這個使用者來跑 bind，會因其無法正確寫入此簽署過的檔，而在後續的過程中有錯誤，故應該將檔案 owner 修改，以 Ubuntu 來說，應執行以下指令

```
root@Ubuntu:~$ chown -R bind /etc/bind/example.com
```

5) 解凍 zone 使其生效

做完手動修改後，必須將其解凍，BIND 會重新載入 zone file，後續網域自動維護的功能也會恢復運作。指令如下：

```
root@Ubuntu:~$ rndc thaw
```

以上五個步驟當中，事實上第二到第五個步驟的指令相當固定且單純，管理者可以寫一個 script 將其整合，降低管理負擔。

4.2. 緊急降級為 DNS 伺服器

DNSSEC 的運作有賴一連串金鑰與簽章的驗證，如果 DNSSEC 受攻擊等原因而不可使用，其安全設計也會讓 DNSSEC resolver 不接受純 DNS 的回應，避免遭到詐騙。

然而若是管理失當而造成 DNSSEC 不可使用，最佳的解決方法是手動降回 DNS，先讓外部的 resolver 仍能解析本網域，之後再慢慢釐清 DNSSEC 的問題。

手動降回 DNS 的方法，即為要求上層網域將本網域的 DS record 給移除，由於本網域的 DS record 不存在，信任鏈中斷，因此 DNSSEC resolver 也不會強制要求本網域的金鑰與簽章的正確性。

特別要注意的是，即使上層網域的 DS 已移除，但仍可能被 cache 住而影響一段時間，所以最好在佈署 DS 當時，即將 DS 的 TTL 設的比較小(建議為 300)，移除時可較快生效。

4.3. DNSSEC 伺服器遭入侵時處理

一般來說，以 2048 bit 金鑰的強度，如果要用暴力法在時限內破解，就算是國家級的超級電腦也不易在短時間內達成。被破解通常是

DNSSEC 伺服器被直接入侵取得金鑰，或者管理人員的問題，這是維護 DNSSEC 伺服器必須要注意的。

我們建議若是伺服器遭到入侵，較簡單的作法是先依照 4.2 節的作法，降成 DNS 伺服器，然後依照第 3 章的步驟，創造新的金鑰並重新建立起 DNSSEC 伺服器。這邊要注意的是 DNSSEC 伺服器重新上線的速度不能快過於原 zone file 中最大 TTL 的時間，避免外部 DNSSEC resolver 還 cache 住舊的金鑰，卻取到新的 RRSIG 而產生驗證錯誤的情況。

另外可參考完整版，有另一個較嚴謹但比較複雜的方法，可僅更換金鑰而不需重建整個 DNSSEC 伺服器，如果確定伺服器沒有被入侵，但有更換金鑰的需求的話，也可採用完整版的作法。

5. 除錯資訊

5.1. 檢查設定檔

常見 BIND 無法正確運作的原因是設定檔錯誤，有可能是打字錯誤或重複設定等情況。可利用指令 named-checkconf 來驗證設定檔是否有錯，也可利用 named-checkzone 來驗證 zone file 是否有錯。

5.2. 設定適當的時間值

在 4.2 節描述到，上層 DS 的 TTL 最好設為 300，這樣不管是否被外部的 resolver cache 住，都可以在 5 分鐘內，有效的降為 DNS。而除了上層的 DS 之外，本網域的任何 RR 都建議設為 300，這樣一來雖然會提高流量，但任何的簽章錯誤都可以在 5 分鐘內修正回來，而不致於 cache 過久無法改變。如果擔心流量或修正不了這麼多的 RR，建議 DNSKEY 及重要的 RR(如 SOA，及 www.example.com 的 A record)，一定要將 TTL 降低。

另外 SOA 的 Negative TTL 值也建議設為 300，如此一來，若外部的 resolver 原先 query 不到的 RR，也至少在每 5 分鐘內會重試一次，儘快取得最新值。

為了讓所有的時間都有正確的比較基準，也建議使用 ntp 來校時，讓所有的 master、slave、及 resolver 機器的時間都很精準，否則在簽章的有效期方面可能會出現問題。

5.3. CentOS 與 RHEL

就我們實驗上發現，CentOS 6.2 及 RHEL 6.2 的 nsupdate 指令都有明顯的 bug，會導致 memory dump 而跳出程式，雖然還看不出明確的功能問題，但仍不建議在此兩個平台使用 nsupdate 指令，而本

精簡版 SOP 也沒有用到此指令。

另外 RHEL 6.2 有一個 SOA 的問題。根據標準規定 SOA 的第一個參數應為 master name server 的 FQDN，由於此參數很少被實際利用，所以有可能管理員沒有正確設定而不自知。在我們曾經使用過的 OS 裡面(FreeBSD、Ubuntu、CentOS、RHEL)，我們發現僅有 RHEL 在此參數設定不正確時，Slave 會不回應 RRSIG 而導致錯誤，我們認為這是個 bug，因為 master name server 是否正確設定與否，不應跟 RRSIG 有邏輯上的關係。我們還發現在使用 view 功能時，做 zone transfer 動作有時會出現 delay 的情形。另外還發現 RHEL 6.2 在管理員關閉 process 時，有時會出現無法正常關閉的現象，在此提醒讀者注意。

5.4. 與 OS 層的配合

若管理者使用 Linux 的 SELinux 或 FreeBSD 的 mtree 的時候都要特別注意。這兩個程式都是用來加強安全性，同時也限制了檔案的讀寫，然而 BIND 的工作目錄是經常要被寫入的，因為做為 resolver 時，BIND 會經常性的更新 root 的金鑰，而做為 authoritative server 時，則經常要重新簽署簽章。如果沒有開放 SELinux 或 mtree 的權限，可能導致無法運作，或可以運作但是 CPU 會間歇性的飆高。

5.5. 運作一段時間後出現無法解析的狀況

管理者在執行本 SOP 時，有時在執行步驟上有一些疏漏而導致問題發生。在佈署的過程中，由於有步驟可以驗證佈署的正確與否，所以比較容易發現問題並修正，但即使佈署完成後，仍可能因為設定不正確，使得上線一段時間之後，仍發生無法解析的錯誤。

我們常見有部分網域，在上線一個月之後，無法被外部的

DNSSEC resolver 所解析，原因即為自動重新簽署的設定錯誤。以 BIND 的實作來說，RRSIG 每次簽署的有效期限為一個月，當 RRSIG 到期時，BIND 應自動重新簽署 zone file，若未自動重新簽署，則會導致無法解析的狀況。可檢查下列兩項設定是否有錯誤：

1) 檔案/目錄的使用者權限設定錯誤

起因於 named 執行者沒有 zone 目錄與目錄內檔案(金鑰、zone file 等)的寫入權。於本 SOP 3.5 節提到根據採用的作業系統與 BIND 安裝方式的不同，named 的執行者可能不同。以 Ubuntu 來說，named 的執行者為 bind，若 zone file 與金鑰是以 root 身分產生，會因為使用者權限問題使 BIND 無法對 zone file 自動重新簽署造成 RRSIG 過期。

修正方式請參照本 SOP 3.5 節，將 zone 目錄與目錄內檔案(金鑰、zone file 等)的 owner 更改為 named 執行者。

2) 沒設定 update-policy local 參數

於本 SOP 3.2 節提到除了設定 auto-dnssec maintain 參數之外，需搭配設定 update-policy local 參數，否則 named 會無法啟動或 DNSSEC 會因 RRSIG 過期而無法運作。起因於 BIND 無開啟 local 端更新權限，auto-dnssec 無更新權限而無法自動重新簽署 zone file，造成 RRSIG 過期。

修正方式請參照本 SOP 3.2 節，於 named.conf.local 設定檔加入 update-policy local 參數。

5.6. Zone file 簽署後 Slave 機器出現無法同步的狀況

管理者在執行本 SOP 時，若有發生 slave 機器於 zone file 簽署後無法同步的狀況，可檢查下列兩項設定是否有錯誤：

1) 檔案/目錄的使用者權限設定錯誤

起因於 slave 機器的 named 執行者沒有 zone 目錄與目錄內檔案的寫入權，讓 slave 機器於同步時無法將資料寫入。修正方式請參照本 SOP 3.5 節，於 slave 機器把 zone 目錄與目錄內檔案的 owner 更改為 named 執行者。

2) master 機器的 zone file 沒有 slave 機器的 NS 與 A 資訊

本 SOP 作法為當 master 機器簽署 zone file 後會主動通知 slave 機器進行同步。master 機器要主動通知 slave 機器同步時，會先檢視 zone file 的 NS 記錄並判斷哪些伺服器為 slave 機器，最後送出通知。因此，需檢查 master 機器的 zone file 是否有 slave 機器的 NS 與 A 資訊存在。

5.7. 利用 sigchase 參數檢查信任鏈是否正確

如果 root → .com → .example.com 之間任一網域出現問題，都可能造成 www.example.com 無法正確解析，此時可用 dig 指令搭配 sigchase 參數做更深入的分析。sigchase 參數會依序列出從 root 至目標網域所有的驗證記錄，直到發生第一個驗證錯誤發生為止，使用者可藉此記錄找出問題網域。

以下為驗證步驟：

1) 取得 root DNSKEY

sigchase 需要 root DNSKEY 當做信任鏈的起點，因此我們用如下指令把它取回來存成檔案，放在現有目錄下。

```
root@Ubuntu:~$ dig +nocomments +nostats +nocmd
+noquestion -t dnskey . > trusted-key.key
```

2) 開始追蹤信任鏈

檢查的指令如下：

```
root@Ubuntu:~$ dig +topdown +sigchase +multiline -t
a www.example.com
```

相關參數說明：

- +sigchase：追蹤整個信任鏈的驗證過程。
- +topdown：想從 root 開始往 www.example.com 方向檢測時加上此參數，反之，想從 www.example.com 開始往 root 方向檢測時就不要加此參數。

root 至目標網域的信任鏈若是正確的，最後會出現下列訊息：

```
FINISH : we have validate the DNSSEC chain of trust: SUCCESS
```

若最後出現 fail 訊息，就可以從驗證記錄來判斷問題網域以及錯誤原因。

由於此檢測方式遇到第一個錯誤就會停止，但我們並無法確定出問題的網域只有一處，所以這裡建議 topdown 參數加與不加各測一次，藉由從 root → 目標網域以及從目標網域 → root 的驗證，依兩種驗證結果得到的問題網域是否一致來判斷問題網域為一處或兩處以上。

雖然 sigchase 參數可以協助我們追蹤信任鏈的驗證過程，但到目前最新版 BIND 9.9.1 為止，BIND 的實作似乎還有些缺陷，有時使用 sigchase 參數時，dig 程式會有無法執行完畢的情況發生，但絕大部分使用上並不會出現錯誤。

5.8. 第三方 DNSSEC 驗證工具

上節利用 BIND 提供的 dig 指令加 sigchase 參數來驗證信任鏈，在封閉式網路裡，sigchase 參數應是驗證信任鏈的最好方法，但對實際與國際接軌運作的網域而言，有更容易使用的第三方 DNSSEC 驗證工具可選擇。本節將介紹兩種驗證工具，有 DNSSEC Analyzer 與 DNSViz，來協助管理者確認 DNSSEC 功能是否正確運作。DNSSEC Analyzer 能迅速得知驗證結果，而 DNSViz 以圖形介面顯示信任鏈驗證過程，能以更直覺的方式協助管理者除錯。

- DNSSEC Analyzer

  http://dnssec-debugger.verisignlabs.com/

  網頁會以表格方式列出信任鏈驗證結果，如下圖：

優點：

1) 分析速度較 DNSViz 快

2) 若只想知道驗證結果是否正確，從表格裡的 icon 即能迅速得知結果

缺點：

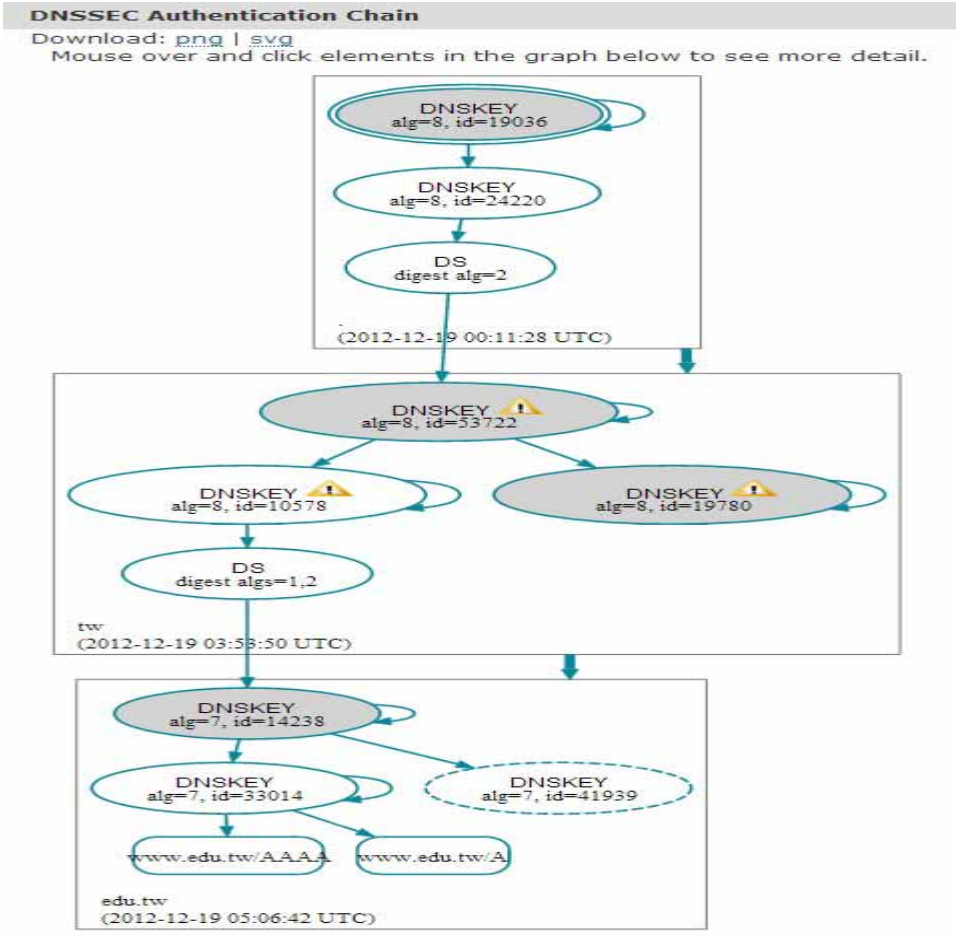
1) 驗證細節以條列式顯示，與 DNSViz 圖形介面相比較不直覺

2) DNSKEY/DS 的資訊不如 DNSViz 詳細

- DNSViz

  http://dnsviz.net/

  網頁會以圖表顯示信任鏈驗證結果，如下圖：



優點：

1) 驗證細節以圖形介面顯示，能迅速理解驗證過程
2) 將滑鼠游標移至 DNSKEY/DS icon 即能得知相關詳細資訊

缺點：

1) 分析速度較 DNSSEC Analyzer 慢
2) 因圖形介面需要較大版面，且需理解圖形所代表的意義，故無法像 DNSSEC Analyzer 能迅速得知驗證結果是否正確