Les logiciels pour vérifier les annonces de routes, avec RPKI et ROA

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 4 février 2012

https://www.bortzmeyer.org/rpki-tests.html

Un ensemble de RFC vient de sortir <https://www.bortzmeyer.org/securite-routage-bgp-rpki-roa.html>, décrivant un mécanisme pour sécuriser les annonces de route sur l'Internet (actuellement, on peut annoncer à peu près ce qu'on veut avec BGP...). Quels sont les logiciels existants qui permettent de regarder ces annonces et/ou de les vérifier ?

Avant de voir l'offre logicielle, il faut rappeler en deux mots comment fonctionne ce système de sécurisation <https://www.bortzmeyer.org/securite-routage-bgp-rpki-roa.html>. Les détenteurs de **ressources** Internet (une ressource est typiquement un préfixe d'adresses IP) obtiennent des certificats numériques prouvant qu'ils sont titulaires de cette ressource. L'ensemble de ces certificats et des entités qui les émettent (typiquement les RIR et LIR) forment la RPKI, "*Resource Public Key Infrastructure*". Muni de son certificat, le titulaire peut émettre des **ROA** ("*Route Origin Authorization*"), des objets signés avec la clé du certificat, qui déclarent quel AS a le droit d'annoncer quel préfixe. L'idée est que les routeurs BGP, recevant une annonce, vont vérifier s'il existe un objet signé authentifiant cette annonce. Sinon, ils peuvent décider de rejeter l'annonce ou bien baisser sa priorité.

Ça, c'est le principe. En pratique, pour que le système passe à l'échelle, pour éviter que les routeurs ne passent leur temps à de coûteuses opérations cryptographiques, une nouvelle classe de machine est introduite, les caches de validation. Ce sont typiquement des serveurs Unix, installés près des routeurs (probablement un cache par POP), qui recevront les ROA, les valideront et diront juste au routeur quelles sont les annonces acceptables. (Ce modèle de déploiement n'est pas le seul obligatoire mais ce sera probablement celui utilisé au début.)

Il y a donc plusieurs classes de logiciels en jeu :

— Les logiciels permettant d'émettre les certificats de la RPKI. Ils seront sans doute analogues à ce qu'utilisent les autorités de certification actuelles. Dans le futur, ils seront probablement intégrés aux logiciels d'IPAM. Je n'ai pas d'expérience avec de tels logiciels. Notez qu'un certain nombre d'acteurs (par exemple le RIPE-NCC) proposent des interfaces Web permettant de leur sous-traiter la gestion des certificats et des ROA.

— Les logiciels des caches de validation, permettant de recevoir les objets signés et de dire lesquels sont valides et lesquels ne le sont pas.
— Les logiciels à bord des routeurs, intégrés au système d'exploitation dudit routeur, et permettant de communiquer avec le cache de validation.

Pour la première catégorie, je l'ai dit, je n'ai pas d'expérience. Il existe des logiciels spécialisés comme celui du RIPE <https://labs.ripe.net/Members/AlexBand/local-certification-service-launched

Pour la seconde catégorie, les logiciels pour gérer un cache de validation, il existe deux importantes suites logicielles sous une licence libre, les outils du RIPE-NCC <http://www.ripe.net/lir-services/resource-management/certification/> et rcynic <http://rpki.net/>. On peut aussi, pour certaines fonctions simples, utiliser des outils existants comme OpenSSL. Enfin, il existe des interfaces Web ou whois permettant de parler à des caches de validation publics.

Voyons d'abord la suite du RIPE-NCC. Elle est disponible en ligne <http://www.ripe.net/lir-services/resource-management/certification/tools-and-resources>. Elle est écrite en Java et on récupère donc un code binaire qu'on peut exécuter directement si on a un JRE. (Les exemples ci-dessous ont été faits avec la version 1, la version 2 a changé bien des choses mais est trop récente pour que j'ai eu le temps de la tester.)

```
% certification-validator -h
usage: For detailed usage scenarios see README file. Options:
...
 -p,--print              Show the certificate repository object in a
...
 -v,--verbose            Show all validation steps
    --version            Show version information
```

Mais le logiciel n'est rien sans les données. Il faut aussi télécharger les objets de la RPKI. Suivant le RFC 6481 [1], le dépôt est accessible en rsync. Ainsi :

```
% rsync -av rsync://rpki.ripe.net/repository RPKI-repository
```

va installer sur votre disque dur tous les objets existants (fin 2011, sur le dépôt du RIPE, ils sont environ 3 000, des statistiques publiques <http://certification-stats.ripe.net/> sont disponibles). Une fois que c'est fait, on peut les afficher. D'abord, un certificat :

```
% certification-validator  --print -f \
     RPKI-repository/64/71d07d-c9f8-4368-87ea-162d1075dfaa/1/q3C8zU6hYpb7zVqmpjlVY5B_BWE.cer
Serial: 85871
Subject: CN=q3C8zU6hYpb7zVqmpjlVY5B_BWE
Not valid before: 2010-12-14T15:26:48.000Z
Not valid after:  2011-07-01T00:00:00.000Z
Resources: 85.118.184.0/21, 93.175.146.0/23, 2001:7fb:fd02::/47
```

Ensuite, une liste de révocations :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc6481.txt

```
% certification-validator  --print -f \
      RPKI-repository/42/a01f82-9193-4199-95a8-83ae5d9c832f/1/vIp8GBZHR75QzAcTIHtf0IcdB3g.crl
CRL type: X.509
CRL version: 2
Issuer: CN=bc8a7c18164747be50cc0713207b5fd0871d0778
Authority key identifier: vIp8GBZHR75QzAcTIHtf0IcdB3g=
Number: 629
This update time: 2011-09-21T14:44:51.000Z
Next update time: 2011-09-22T14:44:51.000Z
Revoked certificates serial numbers and revocation time:
```

Le principal service de ce système de sécurité est de valider les ROA. Ces ROA sont distribués via rsync comme ci-dessus mais aussi disponibles en ligne <https://certification.ripe.net/certification/public/all-roas>. Affichons-en un :

```
% certification-validator  --print -f  roa-5574190.roa
Content type: 1.2.840.113549.1.9.16.1.24
Signing time: 2011-01-11T19:04:18.000Z
ASN: AS559
Prefixes:
    193.5.26.0/23 [24]
    193.5.152.0/22 [24]
    193.5.168.0/22 [24]
    193.5.22.0/24
    193.5.54.0/23 [24]
    193.5.58.0/24
    193.5.60.0/24
    193.5.80.0/21 [24]
```

On voit donc que tous ces préfixes peuvent être légitimement annoncés par l'AS 559. Mais avant de conclure cela, encore faut-il vérifier que le ROA est valide. On peut le valider en ligne <https://certification.ripe.net/certification-validator/>. En local, il faut indiquer au validateur les clefs de confiance à utiliser (typiquement, les certificats des RIR, à partir desquels se fait toute la validation). Ces clés ("*trust anchors*") sont accessibles en ligne. Une fois récupérées, on peut valider, ici avec la clé du RIPE :

```
% certification-validator  -t tal/ripe-ncc-root.tal \
       -f roa-5574190.roa --output-dir .
17:11:44,179 INFO  rsync://rpki.ripe.net/ta/ripe-ncc-ta.cer is VALID
17:11:45,587 INFO  rsync://rpki.ripe.net/ta/ripe-ncc-ta.crl is VALID
17:11:45,593 INFO  rsync://rpki.ripe.net/ta/u75at9r0D4JbJ3_SFkZXD7C5dmg.cer is VALID
17:11:46,975 INFO  rsync://rpki.ripe.net/repository/33/36711f-25e1-4b5c-9748-e6c58bef82a5/1/u75at9r0D4JbJ3_SFkZX
17:11:46,978 INFO  rsync://rpki.ripe.net/repository/33/36711f-25e1-4b5c-9748-e6c58bef82a5/1/Uz_7R2AURj1tzQ3QylGl
17:11:48,183 INFO  rsync://rpki.ripe.net/repository/29/c8c6f6-595c-45b1-a76e-bd192676c9bb/1/Uz_7R2AURj1tzQ3QylGl
17:11:48,186 INFO  file:/home/bortzmeyer/tmp/roa-5574190.roa is VALID
```

Le logiciel a affiché toute la chaîne de validation, de la clé du RIR, jusqu'au ROA, en passant par tous les certificats intermédiaires.

La version 2 de l'outil du RIPE semble avoir supprimé les outils en ligne de commande (?) mais apporte un serveur Web qui écoute par défaut sur le port 8080 (il y a aussi un serveur RTR, cf. RFC 6810) et permet de gérer les ROA et leur validation. Par exemple, on peut définir des listes blanches de préfixe qui seront acceptés même en cas d'erreur de validation (un problème qui sera sans doute fréquent au début). On peut récupérer les anciennes versions du logiciel sur le serveur du RIPE <https://certification.ripe.net/content/public-repo/releases/net/ripe/rpki-validator/rpki-validator-app/>.

Pour gérer un cache de validation, ou tout simplement pour explorer la RPKI ou déboguer des problèmes de routage, une alternative aux outils du RIPE-NCC est rcynic <http://subvert-rpki.hactrn.net/>. On ne peut pas dire qu'il brille par la convivialité de sa documentation, ni par le support disponible, mais il marche très bien. La compilation (rcynic est écrit en C et en Python) à partir du dépôt Subversion est simple (`./configure && make && sudo make install`), mais il ne faut **pas** utiliser l'option `--with-system-openssl` de `configure` en raison de la bogue Debian citée plus bas et aussi parce que rcynic nécessite un OpenSSL très récent (sinon, vous aurez à la compilation des erreurs concernant `v3_addr_validate_path`). rcynic s'installe ensuite dans une "*jail*". Le script d'installation a fait une partie du travail mais pas tout (testé sur une machine Ubuntu). rcynic est livré avec de nombreux programmes mais peu d'explications de leur rôle. `rpkid` est pour les LIR (ou RIR), qui émettent des certificats (première catégorie de logiciels, dans ma classification plus haut). Pour les opérateurs réseau qui gèrent des routeurs, il faut utiliser `rcynic`.

Pour terminer la configuration de la "*jail*", en `/var/rcynic/lib`, j'ai dû faire :

```
# Pour lancer facilement les programmes dans la prison
# aptitude install chrootuid

# Installation des bibliothèques dynamiques dans la prison
# cp /lib/libpopt.so.0* /lib/i386-linux-gnu/libacl.so.1* /lib/i386-linux-gnu/libattr.so.1* /lib/i386-linux-g

# cp /etc/nsswitch.conf /var/rcynic/etc
# Sinon, plein de messages "Host unknown"
```

La grande majorité des problèmes de configuration de la "*jail*" vient de la résolution de noms. Pour tester, copier ping dans la "*jail*", puis `chmod u+s /var/rcynic/bin/ping` puis enfin :

```
% sudo  chrootuid /var/rcynic rcynic /bin/ping www.google.com
ping: unknown host www.google.com
```

Ici, il y a un problème, il faut donc vérifier que les fichiers indispensables ont bien été copiés, comme indiqué plus haut.

Ensuite, on édite `/var/rcynic/etc/rcynic.conf` si on veut changer des paramètres. rcynic récupère lui-même les objets de la RPKI. La première fois, c'est très long (cela peut prendre plusieurs heures). Ensuite, on doit juste exécuter la commande de temps en temps, par exemple depuis cron :

```
# chrootuid /var/rcynic rcynic /bin/rcynic -s -c /etc/rcynic.conf
rcynic[18606]: Finished, elapsed time 3:03:54
```

Une fois qu'on a les données, on peut examiner un ROA :

```
% print_roa RPKI-repository/ff/cf59fc-653f-44eb-aca1-c5d1f27b532d/1/tyIST2Kt6P8tIxIthqPHxCj6lSY.roa
Certificates:   1
CRLs:           0
SignerId[0]:    b7:22:12:4f:62:ad:e8:ff:2d:23:12:2d:86:a3:c7:c4:28:fa:95:26 [Matches certificate 0] [signing
eContentType:   1.2.840.113549.1.9.16.1.24
version:        0 [Defaulted]
asID:           15533
 addressFamily: 1
     IPaddress: 62.73.128.0/19
     IPaddress: 213.212.64.0/18
 addressFamily: 2
     IPaddress: 2001:4138::/32
```

ou bien chercher un ROA en fonction d'un préfixe (chose que je n'ai pas trouvé le moyen de faire avec les outils du RIPE) :

```
% find_roa /var/rcynic 193.5.26.0/23
ASN 3303 prefix 193.5.26.0/23 ROA /var/rcynic/data/authenticated.2011-10-04T09:48:34Z/rpki.ripe.net/repository/2
ASN 559 prefix 193.5.26.0/23 ROA /var/rcynic/data/authenticated.2011-10-04T09:48:34Z/rpki.ripe.net/repository/29

% ./find_roa /var/rcynic 62.73.128.0/19
ASN 15533 prefix 62.73.128.0/19 ROA /var/rcynic/data/authenticated.2011-10-04T09:48:34Z/rpki.ripe.net/repository
```

Notez qu'il existe deux ROA pour le premier préfixe, car deux AS sont autorisés à l'annoncer. Notez aussi que les ROA se trouvent dans le répertoire `authenticated`. rcynic valide les objets au fur et à mesure de leur récupération et son programme `find_roa` ne cherche que parmi les objets validés (les clés de confiance sont livrées avec rcynic, il n'est pas nécessaire de les récupérer explicitement). Si on veut plus de détails sur les ROA trouvés par `find_roa`, on peut appeler `print_roa` sur le fichier trouvé :

```
% print_roa /var/rcynic/data/authenticated.2011-10-04T09:48:34Z/rpki.ripe.net/repository/ff/cf59fc-653f-44eb-aca
Certificates:   1
CRLs:           0
SignerId[0]:    7e:95:45:32:83:04:a1:5e:fa:04:a5:fb:82:28:36:d6:ff:8c:0e:db [Matches certificate 0] [signingTime
eContentType:   1.2.840.113549.1.9.16.1.24
version:        0 [Defaulted]
asID:           15533
 addressFamily: 1
     IPaddress: 176.62.128.0/21
     IPaddress: 62.73.128.0/19
     IPaddress: 213.212.64.0/18
 addressFamily: 2
     IPaddress: 2001:4138::/32
```

Et comment trouver, non pas le (ou les) ROA mais le certificat correspondant à un préfixe IP donné ? La seule méthode que je connaisse consiste en un examen systèmatique du dépôt (ici, on cherche le titulaire de `98.128.4.0/24`) :

```
% for file in $(find /var/rcynic/data/authenticated.2011-10-31T13:36:08Z -name '*.cer'); do
     RESULT=$(openssl x509 -inform DER -text -in $file | grep 98.128.4)
     if [ ! -z "$RESULT" ]; then
        echo $file
     fi
done
/var/rcynic/data/authenticated.2011-10-31T13:36:08Z/rgnet.rpki.net/rpki/rgnet/635/hAK3Zfkl-_L6YZNsUM2T3i8GeVs.ce
```

Et hop, `/var/rcynic/data/authenticated.2011-10-31T13:36:08Z/rgnet.rpki.net/rpki/rgnet/635/L6YZNsUM2T3i8GeVs.cer` contient le certificat convoité.

Si vous voulez tester votre cache validateur, le RIPE annonce des routes avec des ROA <`https://labs.ripe.net/Members/markd/routing-certification-beacons`>, dont certains délibérement invalides, pour tester.

Après ces deux paquetages, développés spécialement pour la RPKI, les outils génériques. Si on préfère utiliser les outils traditionnels, OpenSSL permet de voir les certificats de la RPKI (ils sont normalisés dans le RFC 6487, et utilisent la syntaxe standard de X.509) :

```
% openssl x509 -inform DER -text \
    -in ./RPKI-repository/64/71d07d-c9f8-4368-87ea-162d1075dfaa/1/q3C8zU6hYpb7zVqmpjlVY5B_BWE.cer
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 85871 (0x14f6f)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: CN=5XxoZDVUZvaNYJcVATAgTQ2ifLQ
        Validity
            Not Before: Dec 14 15:26:48 2010 GMT
            Not After : Jul  1 00:00:00 2011 GMT
        Subject: CN=q3C8zU6hYpb7zVqmpjlVY5B_BWE
...
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                AB:70:BC:CD:4E:A1:62:96:FB:CD:5A:A6:A6:39:55:63:90:7F:05:61
            X509v3 Authority Key Identifier:
                keyid:E5:7C:68:64:35:54:66:F6:8D:60:97:15:01:30:20:4D:0D:A2:7C:B4

            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
            Authority Information Access:
                CA Issuers - URI:rsync://rpki.ripe.net/ta/5XxoZDVUZvaNYJcVATAgTQ2ifLQ.cer

            Subject Information Access:
                CA Repository - URI:rsync://rpki.ripe.net/repository/1a/08941d-c629-44be-9708-fc9c3753f8cd/1
                1.3.6.1.5.5.7.48.10 - URI:rsync://rpki.ripe.net/repository/1a/08941d-c629-44be-9708-fc9c3753

            X509v3 CRL Distribution Points:
...
```

Mais les extensions du RFC 3779, cruciales pour la RPKI, sont juste affichées en binaire :

```
            sbgp-ipAddrBlock: critical
                0%0.....0....Uv....]..0.....0.... .....
```

Le problème est en fait que, sur ma machine Debian, OpenSSL a bien le code nécessaire mais il n'est pas compilé avec la bonne option. Il faut recompiler avec `./config enable-rfc3779` ce que ne fait hélas pas Debian (vous pouvez insister pour ce changement via la bogue #630790 <http://bugs.debian.org/630790>, notez que la non-prise en compte du RFC 3779 était peut-être due à CVE-2011-4577 <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-4577>.).

Une fois OpenSSL compilé avec les bonnes options, on voit le certificat entier :

```
           X509v3 Certificate Policies: critical
                Policy: 1.3.6.1.5.5.7.14.2

            sbgp-ipAddrBlock: critical
                IPv4:
                  85.118.184.0/21
                  93.175.146.0/23
                IPv6:
                  2001:7fb:fd02::/47
```

Il existe d'autres outils qui affichent le X.509 et qui semblent capable d'afficher proprement du CMS, comme Legion of Bouncy Castle <http://www.bouncycastle.org/> mais je ne les ai pas testés.

Enfin, dernière façon de regarder et de valider les objets de la RPKI, les services hébergés, comme celui de BGPmon <http://www.bgpmon.net/>. Il est accessible entre autres par whois :

```
% whois -h whois.bgpmon.net " --roa 28001 200.7.86.0/23"
0 - Valid
------------------------
ROA Details
------------------------
Origin ASN:       AS28001
Not valid Before: 2011-01-07 02:00:00
Not valid After:  2012-08-05 03:00:00
Trust Anchor:     repository.lacnic.net
Prefixes:         200.7.86.0/23 (max length /24)
                  2001:13c7:7012::/47 (max length /47)
                  2001:13c7:7002::/48 (max length /48)
                  200.3.12.0/22 (max length /24)

% whois -h whois.bgpmon.net " --roa 2801 200.7.86.0/23"
2 - Not Valid: Invalid Origin ASN, expected 28001
```

On voit que l'AS 28001 a le droit d'annoncer `200.7.86.0/23` (le ROA autorise également trois autres préfixes) mais que le 2801 n'a pas le droit.

Dans le même état d'esprit (validateur accessible via le réseau), on peut aussi se servir d'un "*looking glass*" comme celui de LACNIC <http://www.labs.lacnic.net/rpkitools/looking_glass/> ou de leur service REST <http://www.labs.lacnic.net/site/restful-interface-ovlg_EN> (voyez par exemple l'état de l'annonce <http://www.labs.lacnic.net/rpkitools/looking_glass/rest/valid/cidr/200.7.84.0/23/>).

Maintenant, troisième catégorie, il faut aussi du logiciel sur le routeur lui-même, pour interroger le cache validant. Pour les mises en œuvre de BGP libres comme Quagga, on peut utiliser, par exemple, bgp-srx <http://www-x.antd.nist.gov/bgpsrx/>. Le LACNIC a documenté l'état du logiciel, et la façon de s'en servir, pour plusieurs modèles de routeurs : JunOS <http://www.labs.lacnic.net/drupal/rpki-junos-commands>, et Quagga <http://www.labs.lacnic.net/drupal/rpki-with-quagga>. Le RIPE-NCC a produit une documentation <https://www.ripe.net/lir-services/resource-management/certification/router-configuration> pour les Juniper et Cisco.

Si on veut jouer avec un vrai routeur vivant, EuroTransit <http://www.euro-transit.net/> a configuré des routeurs <https://labs.ripe.net/Members/fhibler/rpki-capable-routers> Juniper publiquement accessibles par telnet : `193.34.50.25` et `193.34.50.26`. Utilisateur "rpki", mot de passe "testbed" et le tout est documenté <http://rpki01.fra2.de.euro-transit.net/documentation.html>. Il y a aussi `juniper.rpki.netsign.net` (rpki/testbed).

```
%  telnet  193.34.50.25
Trying 193.34.50.25...
Connected to 193.34.50.25.
Escape character is '^]'.

        RPKI testbed router
        connected to RIPE RPKI Validator (cache server)
        hosted by EuroTransit GmbH

        For further information about this RPKI testbed:
        http://rpki01.fra2.de.euro-transit.net

        Unauthorized access prohibited
        Authorized access only
        This system is the property of EuroTransit GmbH
        Disconnect IMMEDIATELY if you are not an authorized user!
        Contact 'noc@nmc.euro-transit.net' for help.
```

```
lr1.ham1.de (ttyp0)

login: rpki
Password:
Last login: Tue Dec 13 19:31:05 from 213.211.192.18

--- JUNOS 10.3I built 2011-04-05 18:23:14 UTC
rpki@lr1.ham1.de>


[La validation sur ce Juniper]

rpki@lr1.ham1.de> show validation statistics
Total RV records: 1517
Total Replication RV records: 2976
  Prefix entries: 1412
  Origin-AS entries: 1517
Memory utilization: 446892 bytes
Policy origin-validation requests: 13907256
  Valid: 41302
  Invalid: 39656
  Unknown: 13826298
BGP import policy reevaluation notifications: 27490
  inet.0, 27490
  inet6.0, 0

[Les communications avec le cache, suivant le protocole RTR.]

rpki@lr1.ham1.de> show validation session
Session                                 State   Flaps     Uptime #IPv4/IPv6 records
195.13.63.18                            Up       1015   00:53:27 1207/310

rpki@lr1.ham1.de> show validation database origin-autonomous-system 15533
RV database for instance master

Prefix                 Origin-AS Session                                 State   Mismatch
62.73.128.0/19-19          15533 195.13.63.18                            valid
176.62.128.0/21-21         15533 195.13.63.18                            valid
213.212.64.0/18-18         15533 195.13.63.18                            valid
2001:4138::/32-32          15533 195.13.63.18                            valid

  IPv4 records: 3
  IPv6 records: 1

[Notez le maxlength exprimé sous forme de deux chiffres, toujours
identiques ici mais qui ne le sont pas forcément.]
```

Et si on veut un Cisco ? `telnet rpki-rtr.ripe.net`, utilisateur « ripe », pas de mot de passe :

```
rpki-rtr>show ip bgp rpki table
1187 BGP sovc network entries using 104456 bytes of memory
1280 BGP sovc record entries using 25600 bytes of memory

Network             Maxlen  Origin-AS  Source  Neighbor
24.232.0.0/16       32      10318      0       193.0.19.44/8282
31.3.8.0/21         21      5524       0       193.0.19.44/8282
...

rpki-rtr>show ip bgp 62.73.128.0
BGP routing table entry for 62.73.128.0/19, version 1865228
Paths: (2 available, best #1, table default)
  Not advertised to any peer
  Refresh Epoch 1
  12654 50300 15533
    193.0.4.1 from 193.0.4.28 (193.0.4.28)
      Origin IGP, localpref 110, valid, external, best
```

```
      path 58970BAC RPKI State valid
  Refresh Epoch 1
  12654 50300 15533, (received-only)
    193.0.4.1 from 193.0.4.28 (193.0.4.28)
      Origin IGP, localpref 100, valid, external
      path 58970B68 RPKI State valid

[Regardez la dernière ligne, ci-dessus.]
```

Il faudra aussi apprendre à déboguer les annonces invalides. Si on se fie à l'expérience d'un autre système de sécurité, DNSSEC, il y aura des plantages, et souvent. Mais On n'a pas le choix : tant qu'on ne déploie pas pour de bon, ça ne marchera pas.

Autres logiciels que je n'ai pas eu le temps de tester (il existe une liste à peu près à jour <http://www.ripe.net/lir-services/resource-management/certification/tools-and-resources>):

— RPSTIR <http://sourceforge.net/projects/rpstir/>, un validateur/cache

Quelques autres lectures sur ce sujet :

— Pour Cisco, "*BGP Origin Validation*" <http://www.swinog.ch/meetings/swinog21/p/12_Roque_Gagliano_SwissNog21.pdf>, exposé de Roque Gagliano à Swinog. Il décrit le protocole entre le routeur et le validateur (au dessus de SSH) et contient des exemples Cisco IOS.
— Un bon article de Jérôme Durand <http://reseauxblog.cisco.fr/2012/04/23/jai-teste-pour-vous- > en français.
— Une bonne synthèse sur la RPKI <http://bgpmon.net/blog/?p=414>, avec une bonne discussion à la fin.
— Les transparents du "*RPKI Workshop Routing Lab*" à la réunion Nanog de Denver (le même atelier avait été fait à la réunion RIPE de Vienne) : pour JunOS <http://www.nanog.org/meetings/nanog52/presentations/Sunday/110612.nanog-jlab.pdf> et pour Cisco IOS <http://www.nanog.org/meetings/nanog52/presentations/Sunday/110612.nanog-clab.pdf>.
— La documentation officielle <http://www.juniper.net/techpubs/en_US/junos12.2/topics/topic-map/bgp-origin-as-validation.html> pour JunOS.
— Le retout d'expérience de Tim Hoffman <http://blog.hoff.geek.nz/2014/03/03/rpki-on-junos-is-ea > avec JunOS,
— Un état des outils de la RPKI <http://tools.ietf.org/agenda/80/slides/sidr-10.pdf> qui avait été présenté à l'IETF en mars 2011.
— Une intéressante étude quantitative sur une journée de fonctionnement de la RPKI <https://labs.ripe.net/Members/waehlisch/one-day-in-the-life-of-rpki>.
— Déploiement et utilisation de la RPKI au point d'échange de Lyon <http://gblogs.cisco.com/fr-reseaux/2014/04/29/lyonix-montre-lexemple-pour-la-securisation-de-linternet-f >.
— Plein d'information au RIPE-NCC <http://www.ripe.net/lir-services/resource-management/certification>.