New issue

Jump to bottom

Missing files don't result in a Manifest/PP being considered invalid #319

Closed **job** opened this issue on Apr 28, 2020 · 11 comments

job commented on Apr 28, 2020 Contributor

In a MITM scenario where an attacker intercepts and manipulates the rsync channel (for example strategically withholding certain `.roa` files from the view of the validator being attacked), the resulting set of VRPs will be incomplete and can cause severe operational issues.

When a manifest is valid (manifest is parsable, CRL exists is valid (also not expired), and manifest is signed with keys not revoked by the CRL), and references files which do not exist in the repository at hand, the publication point should be considered compromised.

So in the case of APNIC where an End User (self-hosted) RPKI publication point misses a few .roa files, the validator can proceed to consider all data from all RPs it could reach eligible for further validation, except any data from the publication point where files were missing.

In other words: if one or a few files are missing from the repository, the repository should be considered 'down', and no attempt should be made to start guessing what can be salvaged and what not.

To offer an example:

In manifest `1-tcQDnftkRnWbiMhu2cR1-dgmCs.mft`, a number of ROAs and a CRL are referenced:

```
1-tcQDnftkRnWbiMhu2cR1-dgmCs.crl
IM3xMiMU1QChuWMOGgnbDwYolrU.roa
LkKeUPYrfgzjsOIejLjsHGk44cU.roa
LkKeUPYrfgzjsOIejLjsHGk44cU.roa
fIeCcC8KpdJQd-olU2APdxNZkyA.roa
r5vtD4isKhTzGXNGulSnZ7XCS04.roa
r7TSyWn_GbYPjNWvt4r5ewSNAsk.roa
```

If an attacker withholds all but `LkKeUPYrfgzjsOIejLjsHGk44cU.roa`, so the MITM does the equivalent of:

```
cd repository/DEFAULT/3e/01d411-d915-4277-8fe2-76b0dda2bf3e/1/
rm LkKeUPYrfgzjsOIejLjsHGk44cU.roa \
    fIeCcC8KpdJQd-olU2APdxNZkyA.roa \
    IM3xMiMU1QChuWMOGgnbDwYolrU.roa \
    NpI_Uj3OZb6jvwfUTX0-eOk9QV4.roa
```

and the validator does *not* consider `1-tcQDnftkRnWbiMhu2cR1-dgmCs.mft` in its entirety invalid due to one or more missing `.roa` or `.crl` files, in this specific example the remaining ROA will render all BGP announcements equal to or covered under `80.128.0.0/11 invalid`. This results in actual downtime in real networks.

If the manifest (which references a missing `.roa` file in its entirety is ignored or considered invalid), the remaining `r7TSyWn_GbYPjNWvt4r5ewSNAsk.roa` ROA which would've been transformed into VRP `80.128.0.0/11 AS 0` is also ignored. This renders all BGP announcements under that /11 `not-found` rather than `invalid`, which is a preferable situation.

`rpki-client` now has a patch where if a file is referenced the manifest, but is missing (either due to CA operational error, or because of a MITM attack - we can't differentiate the two), that specific manifest is considered invalid. It will depend on the structure of how CA Certificate trees whether this measure is sufficient or whether a single missing file has to result in the entire publication point being considered invalid.

But until we have more information about other attack vectors, I recommend that a manifest as a whole is considered invalid, when it references a non-existing or wrongly checksummed file. The onus is on the CA publication points to publish correct valid RPKI data, the validator's can't be expected to compensate for CA operator errors (or MITM attacks)

An additional check is needed: if one of the .roa files a manifest references is corrupt (the MITM didn't *delete* strategically file, but filled some .roa files with garbage):

```
echo helewioefwefoibnwef > rpki.ripe.net/repository/DEFAULT/3e/01d411-d915-4277-8fe2-76b0dda2bf3e/1/fIeCcC8KpdJQd-olU2APdxNZkyA.roa

...

rpki-client:  ...trace: error:0DFFF08E:asn1 encoding routines:CRYPTO_internal:not enough data
rpki-client: rpki.ripe.net/repository/DEFAULT/3e/01d411-d915-4277-8fe2-76b0dda2bf3e/1/fIeCcC8KpdJQd-olU2APdxNZkyA.roa: RFC 6488: failed CMS parse
```

corrupting that one .roa file results in an incomplete (operationally problematic) set of VRPs:

```
job@anton ~$ grepcidr 80.128.0.0/11 /var/db/rpki-client/openbgpd
        80.128.0.0/12 source-as 3320
        80.128.0.0/11 source-as 0
        80.128.0.0/11 source-as 3320
        80.144.0.0/13 source-as 3320
        80.152.0.0/14 source-as 3320
        80.156.0.0/16 source-as 3320
        80.157.0.0/16 source-as 3320
        80.157.8.0/21 source-as 3320
        80.157.16.0/20 source-as 3320
        80.158.0.0/17 maxlen 24 source-as 34086
        80.159.224.0/19 maxlen 24 source-as 2792
```

In the above eaxmple there should be ~ 19 VRPs in total, `fIeCcC8KpdJQd-olU2APdxNZkyA.roa` contained 8 VRPs.

In summary:

If a manifests references a non-existing file, or if there is a checksum mismatch between the hash described in the manifest and the hash as derived from the .roa file, the RP MUST consider the whole manifest invalid and not produce VRPs with the remaining .roa files.

job commented on Apr 28, 2020 Contributor Author

I'm aware there is an initiative in SIDROPS to at some point produce a 6486bis which (hopefully) offers better guidance what exactly to do, but that may take months or years.

In the meantime I think this security issue should be addressed in some way, "manifests are truth".

partim commented on Apr 28, 2020 — Member

While your proposal should cover most practical cases, it doesn't consider a case where a resource has ROAs published by multiple CAs, either because it is delegated by a parent to multiple child CAs or because it has ROAs published both by the parent and a child.

The most conservative approach to the situation would be to blacklist all resources that have been associated with the corrupt CA and disregard all ROAs covered by these resources. Incidentally, most caches already implement a mechanism to do this through their local exceptions facilities, so essentially they would only need to maintain an internal exception list and apply it together with the user-provided one.

This could be implemented in one of the next Routinator releases. Question is, should we enable this always, by default, or upon user-request only? Both always and by default would require it to go into an 0.8 since that is a breaking change.

👍 1

job commented on Apr 28, 2020 — Contributor, Author

> While your proposal should cover most practical cases, it doesn't consider a case where a resource has ROAs published by multiple CAs, either because it is delegated by a parent to multiple child CAs or because it has ROAs published both by the parent and a child.

Agreed! There is more work to be done.

> The most conservative approach to the situation would be to blacklist all resources that have been associated with the corrupt CA and disregard all ROAs covered by these resources. Incidentally, most caches already implement a mechanism to do this through their local exceptions facilities, so essentially they would only need to maintain an internal exception list and apply it together with the user-provided one.

Yes, I was thinking along the same lines. We may need some normative text to describe how this works exactly so all validators use the same approach.

> This could be implemented in one of the next Routinator releases. Question is, should we enable this always, by default, or upon user-request only? Both always and by default would require it to go into an 0.8 since that is a breaking change.

I recommend secure by default... so yes! make this the default in 0.8 :-)

lukastribus commented on Apr 28, 2020

Maybe this is an extremist view, or maybe cabin fever is settling in, but given the attack vector, why make this configurable at all? What's the use-case for operators to run without this? Will rpki-client be configurable in this regard **@job** or is it always done?

If the currently available options are already confusing to none other than **@job**, is adding yet another option really a good idea?

At the very least, use negative language like `--insecure` (curl argument for ignore certificate errors), as opposed to `--no-check-certificate` (wget argument for doing the same), so that we can transmit the severity of said option (of course those two argument names cannot be used for this, this is just an example in the difference between negative and neutral language).

👍 1

partim commented on Apr 29, 2020 — Member

Your view isn't extremist at all, I guess I am just overly cautious. So lest someone can provide a good argument why the outlined behaviour might break things, we'll make 0.8.0 behave in that way.

👍 2

job commented on May 3, 2020 • edited — Contributor, Author

> I was thinking along the same lines. We may need some normative text to describe how this works exactly so all validators use the same approach.

An effort is underway here: https://tools.ietf.org/html/draft-ymbk-sidrops-6486bis-00

I hope the emergent thinking expressed in this github issue aligns with the ietf-ized language in that draft.

job mentioned this issue on May 11, 2020

Add sbgp-ipAddrBlock prefixes to slurm-style 'prefixfilter' when CA PP distrusted cloudflare/cfrpki#40

Closed

wk commented on Sep 1, 2020 — Contributor

I notice that some of the work being currently done on implementing RTA here appears to have opened the door toward a rewrite of elements of the repository parsing and validation functionality.

Is this issue expected to be closed as part of the work on RTA, and are both targeting 0.8?

partim commented on Sep 1, 2020 — Member

Yes and yes. As both RTA and dropping entire CAs needed some refactoring in the validation code, it seemed reasonable to do both at the same time.

👍 1

wk commented on Sep 1, 2020 — Contributor

Awesome! Looking forward to 0.8. Thanks for the super quick reply!

partim commented on Sep 22, 2020 — Member

This was fixed via #317.

partim closed this as completed on Sep 22, 2020

---

job commented on Sep 22, 2020 — Contributor — Author

Small correction: this was fixed in #371

wip-sync pushed a commit to NetBSD/pkgsrc-wip that referenced this issue on Dec 14, 2020

`CVE-2020-17366 was fixed with the routinator 0.8.0 release, ...` b7a1839

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

4 participants