Root Algorithm Rollover and Lab Experiment in BII

Davey Song , Kevin Gong / BII

2019-3-2 / Tokyo / Yeti DNS Workshop

Bii
天地互连

Uncertainties of ECC for DNSSEC

- Impact of switching to ECC on resolvers is uncertain
  - Performance issue, up to an order of magnitude slower than RSA
  - ECC readiness of Resolvers (Large install base) is uncertain especially the Auto-Rollover from RSA to ECC
  - The right timing is vital for success of Algorithm rollover

”the new ZSK wasn’t pre-published long enough” , ”this resulted in validation errors” from Unbound-users mailing list

DNSSEC Algorithm Rollover approach

- Specified in RFC6781 and RFC4035, using double-signature rollover , expect one signature for each algorithm in the zone apex

The conservative approach interprets this section very strictly, meaning that it expects that every RRset has a valid signature for every algorithm signaled by the zone apex DNSKEY RRset**,** including RRsets in caches. **The liberal approach** uses a more loose interpretation of the section and limits the rule to RRsets in the zone at the authoritative name servers.

----section-4.1.4 of RFC6781

- Although RFC6781 recommend conservative approach, many open source signers like BIND "managed keys" and OpenDNSSEC implements the “liberal” approach.

DNSSEC Algorithm Rollover

- Experience provided by practice on level of second domain by RIPE NCC and TLD .BR , .SE ,
  - RIPE NCC suggest to roll both ZSK and KSK (2015)
  - .SE Algo Roll adopted liberal approach with 6 failure out of 10,000 probes (2018)
- There is no existing experience on the level of Root (automatic algorithm rollover for trust anchors, RFC5011 considered)
- It is still interesting and unknown whether ZSK and KSK should be rolled at the same time

Algorithm rollover in Lab Environment

- To test potential configurations as many as possible
  - Both Conservative and liberal approaches
  - Roll KSK without ZSK, and Roll them at the same time
- Four test configurations are proposed
  - **Test1:** Republish KSK without signature as we rolled the key (Yeti KSK rollover), intentional violation of RFC6781
  - **Test2:** Similar with Test1 but republish KSK and its signature without rolling ZSK
  - **Test3:** Roll both ZSK and KSK in liberal approach
  - **Test4:** Roll both ZSk and KSK in conservative approach

Test Setup

- For each test, setup 3 authoritative servers
  - 1 Master : BIND 9.11.5-P1
  - 2 Slave: Knot 2.7.6, NSD 4.1.26
  - Set DNSKEY TTL: 600 seconds
- For each test, setup 2 resolvers
  - BIND 9.11.5-P1， Unbound 1.8.3
- Monitoring setup
  - Check rfc5011 state by recording the managed.key file on two resolvers (managed.key file)
  - Monitor the trust chain by recording the response for random/junk queries to see whether the AD bit is set for a valid response
  - Monitoring the changes of Root zone (DNSKEY record and signature)
  - Capture DNS packet via dnscap on all servers

Fast Algorithm rollover in 10 minutes

- Since RFC5011 timer (wait 30 days to trust a new KSK) is too long, we hack the resolver to accept a shorter timer to get a result in a stand-on time
  - Add Hold-Down Time: 60 second
  - Remove Hold-Down Time : 60 second

Restart Bind9:
named -c /etc/named.conf -t /var/named -u named -T mkeytimers=2/5/60

Edit unbound.conf:
add-holddown:60
del-holddown:60
permit-small-holddown:yes
keep-missing:300

All Tests got passed on fast algorithm rollover!

Test1: Timeline and results

	slot 1	slot 2	slot 3	slot 4	slot 5	slot 6	slot 7	slot 8
old KSK	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	pub	revoke+sign	
new KSK		pub	pub	pub	pub	pub+sign	pub+sign	pub+sign

ZSK inactive

- Original design: 10 days for each slot
- We just wait 30 days to and manually check if the key is trusted in resolver's "*managed.key*" file and the validation status
- An accidental mistakeZSK become inactive in slot 5 before the new key trusted. It resulted validation failure
- During slot 5 SERVFAIL for BIND resolver and No AD bit set in Unbound resolver (with 'val-permissive-mode: yes')
- RFC5011 …OK

Test2: Timeline and result

	slot 1	slot 2	slot 3	slot 4	slot 5	slot 6	slot 7	slot 8
old KSK	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	pub	Revoke+sign	
new KSK		pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign

- Pass the test!
- Both BIND and unbound accept and trust the new key and new algorithm when 30-day timer expires
- The validation tests got passed during the whole process （slot6 , slot 7 and slot 8）

Test3: Timeline and result

	slot 1	slot 2	slot 3	slot 4	slot 5	slot 6	slot 7
old KSK	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	Revoke+sign	
new KSK		pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign
old ZSK	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	
New ZSK		pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign

inactive

- An accidental mistake: KSK become inactive in slot 5 before the new key trusted. It resulted validation failure for both BIND and Unbound resolver
- During slot5 SERVFAIL for BIND resolver and No AD bit set in Unbound resolver (with ‘val-permissive-mode: yes’)
- BIND restart the Add Hold-Down Time for another 30 days
- Unbound continue the timer and trusted the new key after the timer expired

Test4: Timeline and result

	slot 1	slot 2	slot 3	slot 4	slot 5	slot 6	slot 7	slot 8
old KSK	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	Revoke+sign		
new KSK			pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign
old ZSK	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	sign	
New ZSK		sign	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign	pub+sign

- Test4 results the same with Test3 as well as the accidental mistake

Conclusion

- All Tests supposed to be passed if there is no key timing error even for test 1
- Future tests should be done
  - Test roll back if failure observed
  - Test stand-by Key
- Invite more resolvers to join