Key States Explained

As well as access to the actual key pair OpenDNSSEC needs to keep track of the state of a key. It must know if the key is published, used for signing and when it is safe to remove it. Formerly the internal representation would reflect what is described in the RFC DNSSEC Key Rollover Timing Considerations. For OpenDNSSEC users these are the familiar states as ready, active, retire. One of the design goals of OpenDNSSEC 2.0 is to support different kinds of key rollovers. For this the old representation was not expressive enough.

Starting from OpenDNSSEC 2.0 the internal representation of a key is no longer described as one single state machine, but rather four separate state machines. One for every public element of a key.

- DS resource record
- DNSKEY resource record
- The RRSIG over the DNSKEY set
- The RRSIGs over all other resource record sets.

This provides a fine grained state of a key and is able to express any state a key could have in OpenDNSSEC 1.x. However the reverse is not true. Not every combination of states in OpenDNSSEC 2.0 has an equivalent in OpenDNSSEC 1.x. Yet, OpenDNSSEC will still by default present the user with the old lingo to provide a familiar interface in `ods-enforcer key list` (was: `ods-ksmutil key list`). We did our best to provide a mapping as close as possible. However users should realize the internal state might be slightly different than could be assumed from experience with OpenDNSSEC 1.x.

Using `ods-enforcer key list -d` one could list the actual representation of a key in the database.

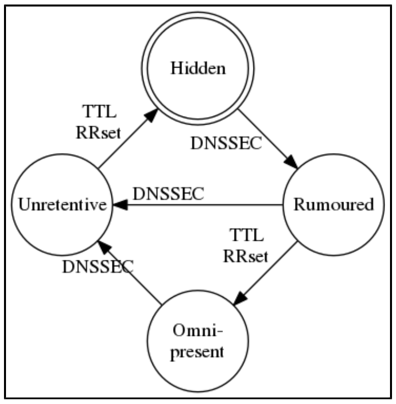
1.x compatible vs. 2.0 key list

```
$ ods-enforcer key list
Zone:        Keytype: State:  Date of next transition:
example.com  KSK      publish 2016-04-15 00:22:18
example.com  ZSK      ready   2016-04-15 00:22:18



$ ods-enforcer key list -d
Zone:        Key role: DS:    DNSKEY:  RRSIGDNSKEY: RRSIG:      Pub: Act: Id:
example.com  KSK       hidden rumoured rumoured     NA          1    1    7a188b4177bed1a744c1bcb9aa4362cf
example.com  ZSK       NA     rumoured NA           omnipresent 1    1    1c86793b7b779d935756906ec7fd28f7
```

In the above example you see a zone with two keys, a KSK and a ZSK. The columns DS, DNSKEY, RRSIGDNSKEY, RRSIG correspond with the state machines mentioned earlier. The Pub, Act columns are derived from these states and show how the Enforcer instructs the Signer to use these keys. Each state machine has four states: hidden, rumoured, omnipresent, and unretentive. NA stands for not applicable and is displayed when the type of key makes no use of this resource record.



A brief description of the states:

- **Hidden**. The associated resource records are never published or are unpublised long enough that the enforcer knows no validator will try to use this resource record for validation.
- **Rumoured**. This record is being published but might not be accessible for all validators yet. After some time (including the TTL of record set) the state will move to omnipresent.
- **Omnipresent**. The record is fully published and available for everyone. No caches contain any old key with an unexpired TTL.
- **Unretentive**. The signer is instructed no longer to publish this record but it might exist in some caches still. After some time the state will move to hidden.

Transitions between states are either motivated by *DNSSEC* or by *TTL*. For example the transition from Omnipresent to Unretentive is only performed when doing so will not prevent the zone from being properly DNSSEC signed. In this transition a resource record will be removed from the zone. The Transition from Rumoured to Omnipresent does not involve publishing extra records and only depend on the TTL (among other timing parameters) of the involved record sets. When the state reaches Omnipresent we know this new information is available for everyone to pick up regardless previous caching actions.

If the Pub flag is set the Signer is instructed include the DNSKEY record associated with this key in the signed zone. The Act flag instructs to use the key to produce signatures (either over all zone data or the DNSKEY set, depending on the type of key).

Users are advised to include the output of `ods-enforcer key list -d` in bug reports and questions regarding timing on keys in OpenDNSSEC 2.0.

DS State at Parent

In addition to the key state, the DS internally has a handful of sub states regarding the publication at the parent side. They can be operated on with the *key ds-** family of commands. The states are respectively (names taken from code):

1. KEY_DATA_DS_AT_PARENT_**UNSUBMITTED**. The DS record does not need to be published.
2. KEY_DATA_DS_AT_PARENT_**SUBMIT**. The Enforcer wants this DS to be uploaded to the parent. It either waits for the user confirming the upload, or when the DelegationSignerSubmitCommand in the KASP is set, executes that command. In both cases it will proceed to the next state.
3. KEY_DATA_DS_AT_PARENT_**SUBMITTED**. The Enforcer knows this DS is to be published soon. It might or might not be visible to the world yet. In the *key list* this will show as *waiting for ds-seen*. An external entity should initiate this *ds-seen*, be it a user or for example a cron job that polls the parent.
4. KEY_DATA_DS_AT_PARENT_**SEEN**. The DS available for everyone. It will stay in this state until the KSK must make place for a new KSK. In which case it will move to the next state automatically.
5. KEY_DATA_DS_AT_PARENT_**RETRACT**. Similar to SUBMIT it will wait for user input or, if available, execute the DelegationSignerRetractCommand. Then moves to the next state.
6. KEY_DATA_DS_AT_PARENT_**RETRACTED**. The key is waiting for a *ds-gone*. Once given the DS state will go back to UNSUBMITTED.