Parallel ZSK/KSK Rollover Scheme

Zheng Wang

wangzheng@conac.cn

China Organizational Name Administration Center (CONAC)

April 10th, 2013

Outline

The problem

- The separation of ZSK and KSK rollover
  - Allow ZSK to rollover more frequently than KSK
  - Believed to simplify the complicated and vulnerable key rollover operations

The problem

- The separation of ZSK and KSK rollover
  - Allow ZSK to rollover more frequently than KSK
  - Believed to simplify the complicated and vulnerable key rollover operations
- Does it really help?
  - Sequential ZSK and KSK rollover takes long time
  - Help little in lowering operational complexity

The problem

- The separation of ZSK and KSK rollover
  - Allow ZSK to rollover more frequently than KSK
  - Believed to simplify the complicated and vulnerable key rollover operations
- Does it really help?
  - Sequential ZSK and KSK rollover takes long time
  - Help little in lowering operational complexity

Emergency rollover when both ZSK and KSK are compromised

Speed is the top priority!

The Solution

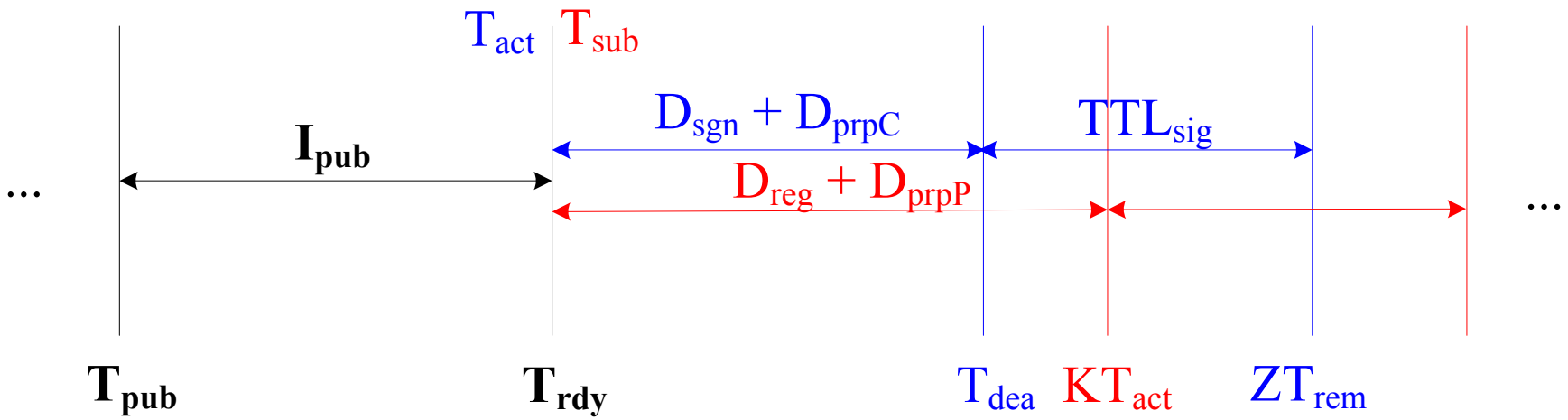
- Parallel ZSK and KSK rollover
  - Enable fast emergency ZSK and KSK rollover
  - Employ similarities between ZSK and KSK rollover algorithms

The Solution

- Parallel ZSK and KSK rollover
  - Enable fast emergency ZSK and KSK rollover
  - Employ similarities between ZSK and KSK rollover algorithms
- The advantage
  - Avoid incurring significant complexity
  - Minimize transition delays

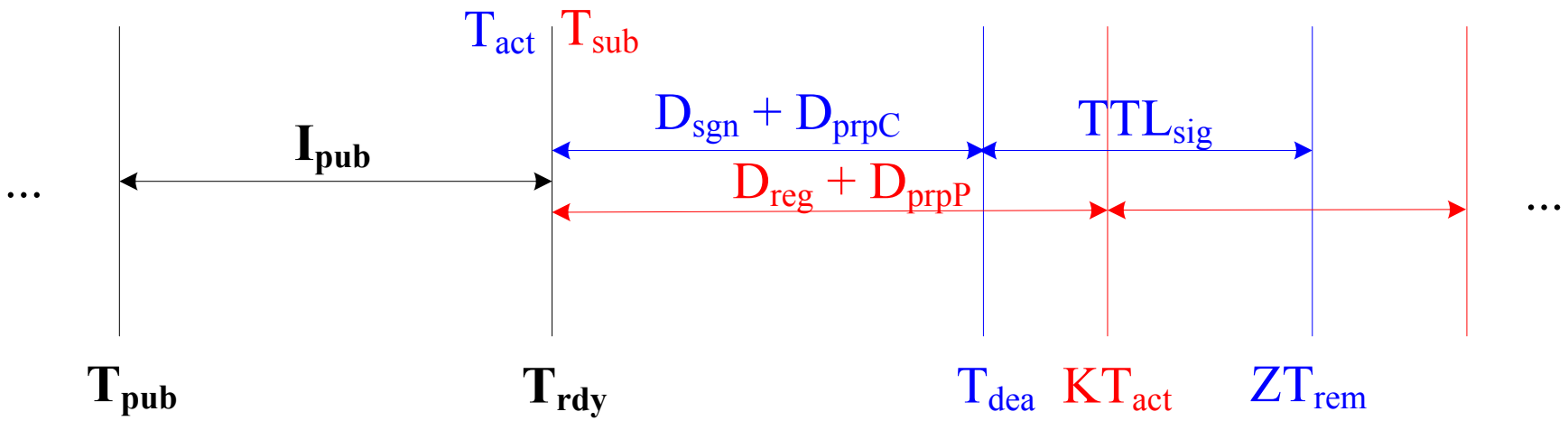
The Scheme

- The time line
  - At least one KSK and one ZSK are active before rollover starts
  - Significant times and time intervals are marked



Event 1

- The successor ZSK and KSK are simultaneously published (Tpub)
  - The successor ZSK and KSK are added to the DNSKEY RRset
  - The new DNSKEY RRset is re-signed by both the current and successor KSK



Event 2

- The publication interval (Ipub)
  - The successor ZSK waits for Ipub before signing the RRset
  - The successor KSK waits for Ipub before submitting to the parent zone

$$Ipub = DprpC + TTLkey \tag{1}$$

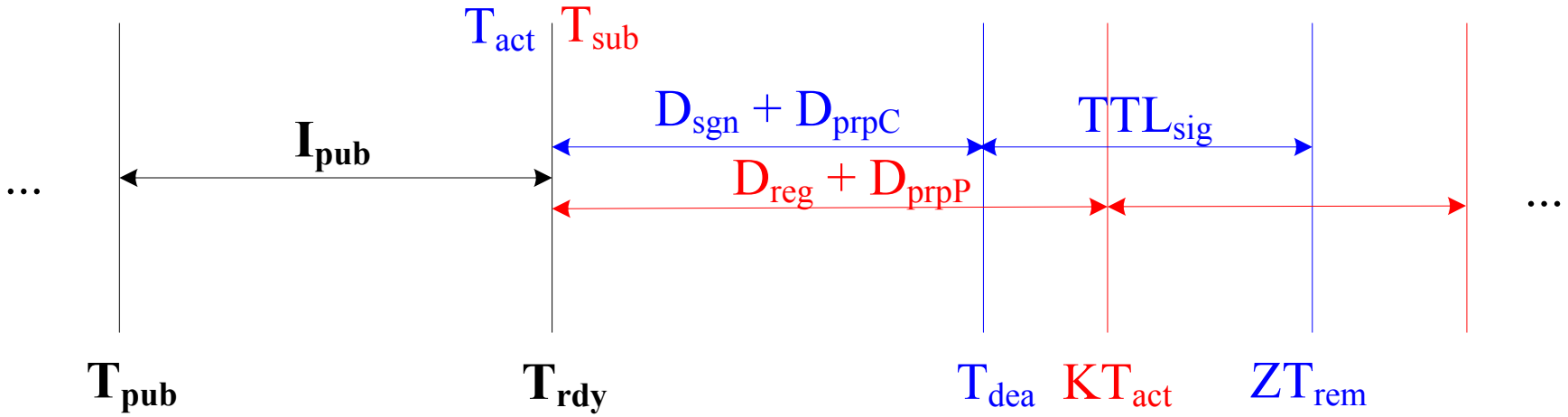
where $DprpC$ is the propagation delay, $TTLkey$ is the time-to-live (TTL) for the DNSKEY RRset

- The key's ready time (Trdy)

$$Trdy = Tpub + Ipub \tag{2}$$

Event 3

- The successor ZSK starts being used to sign RRsets (Tact)
- The DS record corresponding to the new KSK is submitted to the parent zone for publication (Tsub)
- Tact and Tsub can take place simultaneously immediately after Trdy in a bid to minimize delay



Event 4

- For ZSK, all existing RRsets are re-signed and available in all slave servers (Tdea)

$$Tdea = Tact + Dsgn + DprpC \tag{3}$$

where $Dsgn$ is the delay needed to ensure that all existing RRsets have been re-signed with the new key, $DprpC$ is the propagation delay

- For KSK, the DS record is published in the parent zone (KTact)

$$KTact = Tsub + Dreg + DprpP \tag{4}$$

where $Dreg$ is the registration delay, $DprpP$ is the propagation delay for the DS record from the master of the parent zone to replicate to all slaves servers

Event 5

- After the RRSIG records created using the retired ZSK expire from all resolver caches, the retired ZSK can be removed from the zone's DNSKEY RRset(ZTrem)

$$ZTrem = Tdea + TTLsig \tag{5}$$

where $TTLsig$ is the maximum TTL of all the RRSIG records in the zone created with the retired ZSK

- After any caches that contain a copy of the DS RRset have a copy containing the new DS record, the retired KSK is removed from the zone's DNSKEY RRset (KTrem)

$$KTrem = KTact + TTLds \tag{6}$$

where $TTLds$ is the TTL of the DS record

Concluding and Remarks

- A parallel ZSK and KSK rollover scheme with short transition delay and low complexity is proposed
- This rollover delay can be approximated as $DprpC + TTLkey + max\{Dsgn + DprpC + TTLsig, Dreg + DprpP + TTLds\}$
- The scheme can be applied to the emergency ZSK and KSK rollover

Thanks!