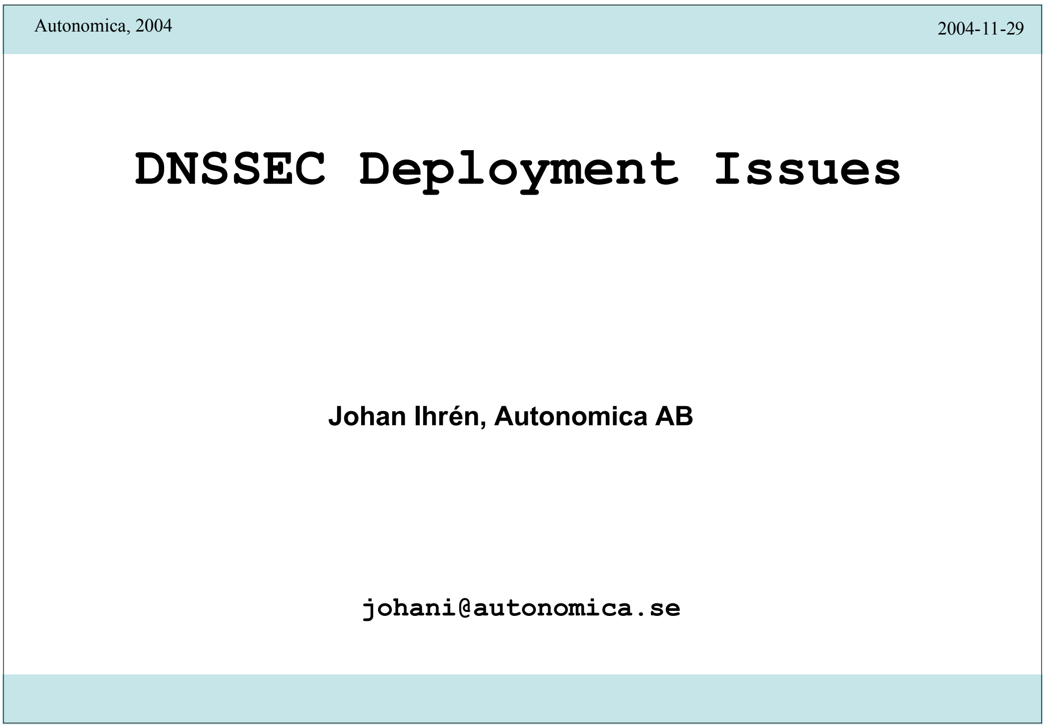
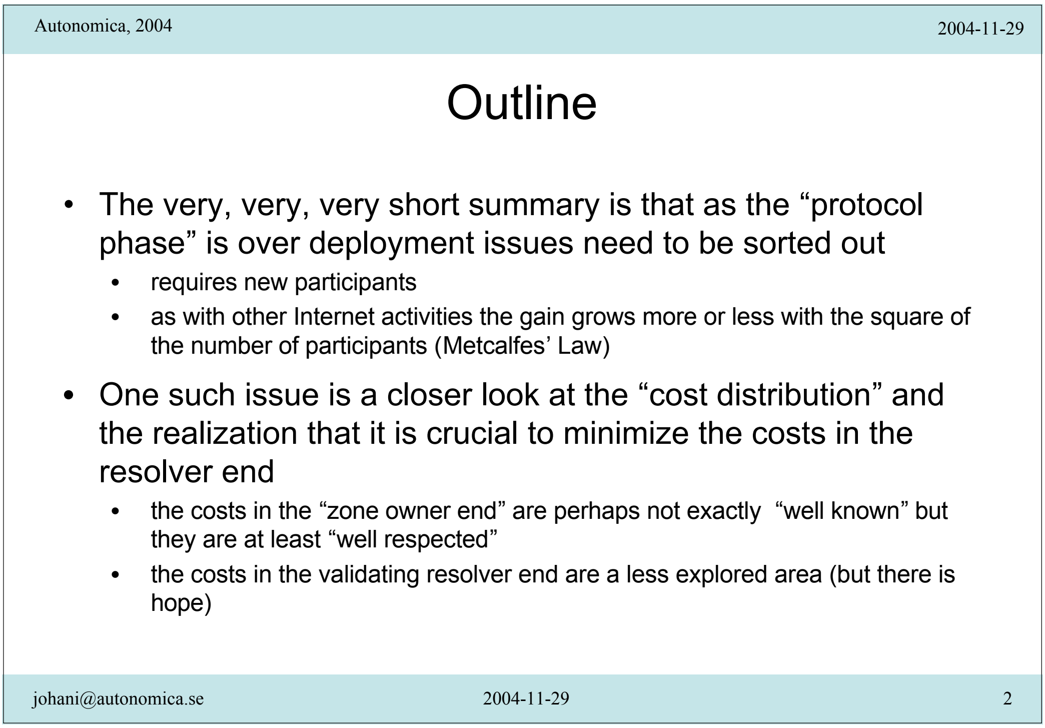


Autonomica, 2004 2004-11-29

DNSSEC Deployment Issues

Johan Ihrén, Autonomica AB

johani@autonomica.se

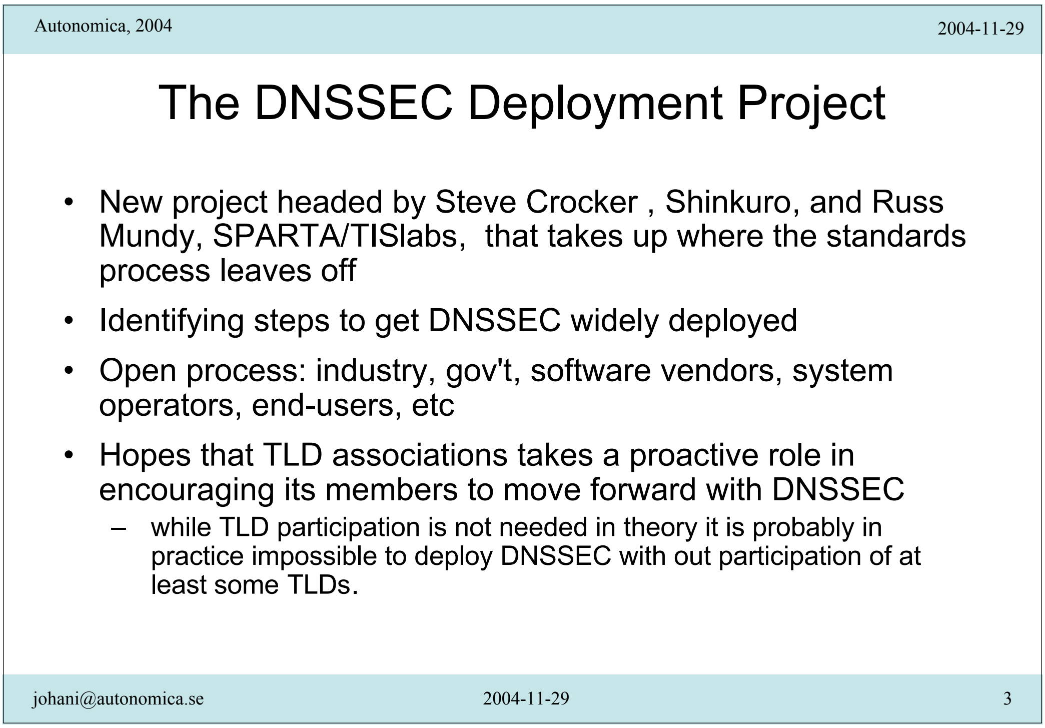


Autonomica, 2004 2004-11-29

Outline

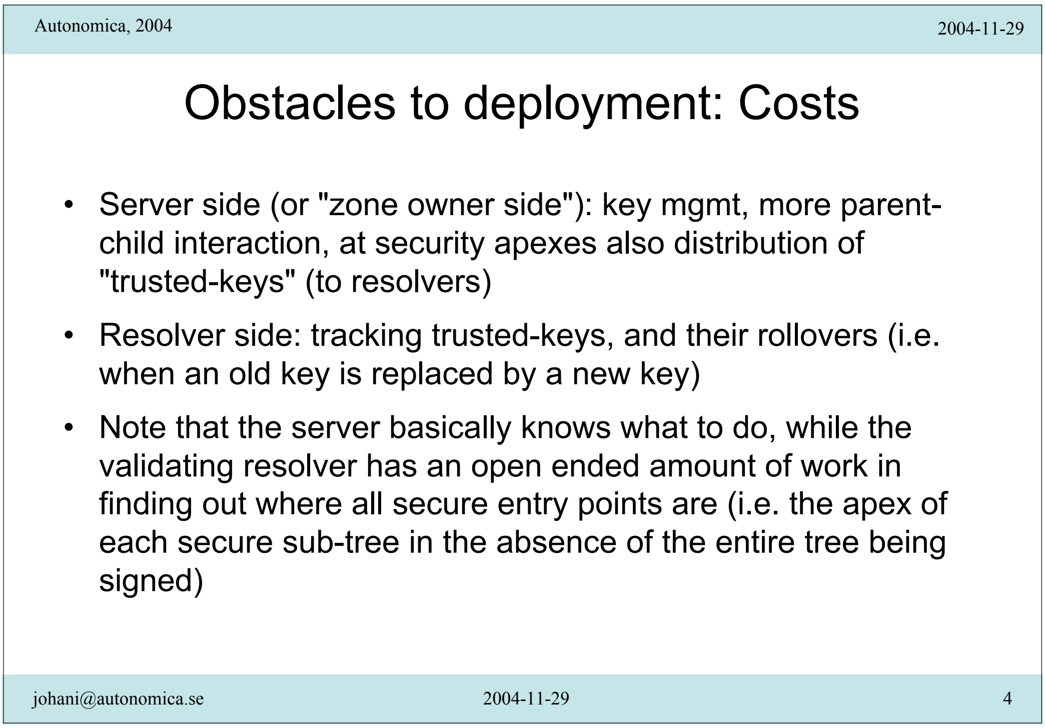
- The very, very, very short summary is that as the “protocol phase” is over deployment issues need to be sorted out
  - requires new participants
  - as with other Internet activities the gain grows more or less with the square of the number of participants (Metcalfes’ Law)
- One such issue is a closer look at the “cost distribution” and the realization that it is crucial to minimize the costs in the resolver end
  - the costs in the “zone owner end” are perhaps not exactly “well known” but they are at least “well respected”
  - the costs in the validating resolver end are a less explored area (but there is hope)

johani@autonomica.se 2004-11-29 2



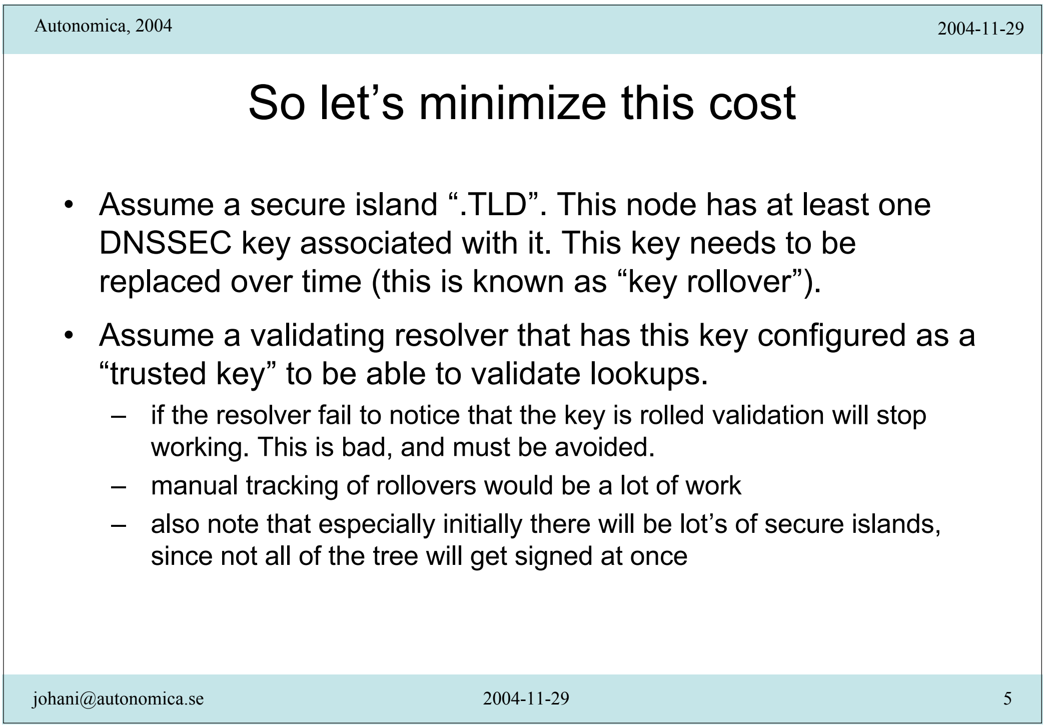
The DNSSEC Deployment Project

- New project headed by Steve Crocker , Shinkuro, and Russ Mundy, SPARTA/TISlabs, that takes up where the standards process leaves off
- Identifying steps to get DNSSEC widely deployed
- Open process: industry, gov't, software vendors, system operators, end-users, etc
- Hopes that TLD associations takes a proactive role in encouraging its members to move forward with DNSSEC
  - while TLD participation is not needed in theory it is probably in practice impossible to deploy DNSSEC with out participation of at least some TLDs.



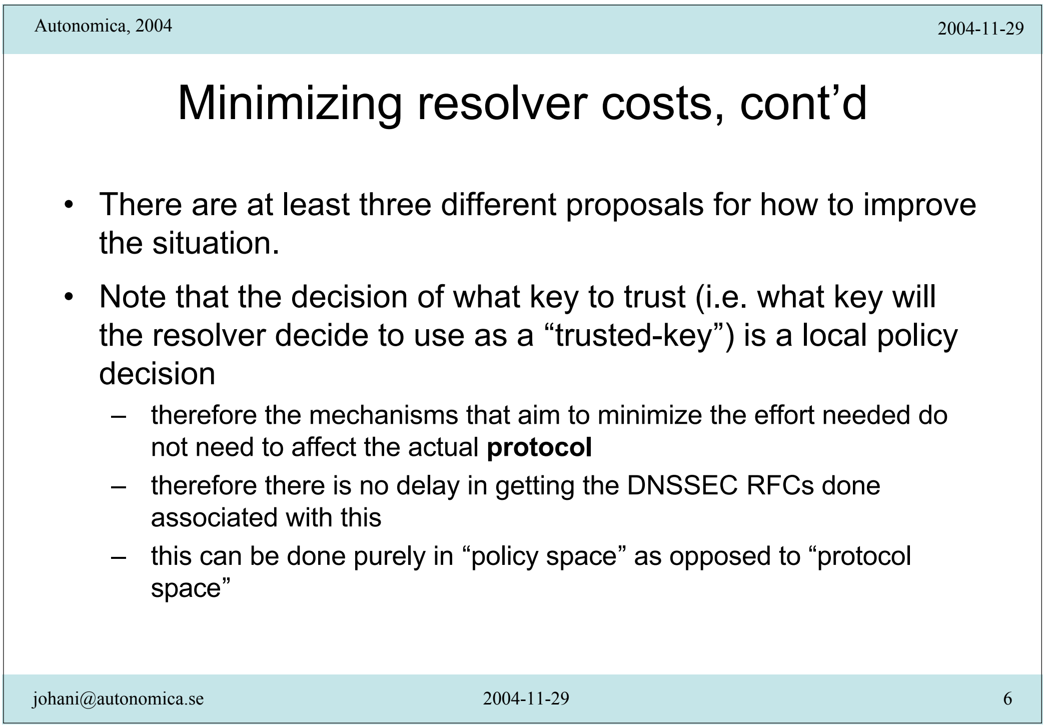
Obstacles to deployment: Costs

- Server side (or "zone owner side"): key mgmt, more parent-child interaction, at security apexes also distribution of "trusted-keys" (to resolvers)
- Resolver side: tracking trusted-keys, and their rollovers (i.e. when an old key is replaced by a new key)
- Note that the server basically knows what to do, while the validating resolver has an open ended amount of work in finding out where all secure entry points are (i.e. the apex of each secure sub-tree in the absence of the entire tree being signed)



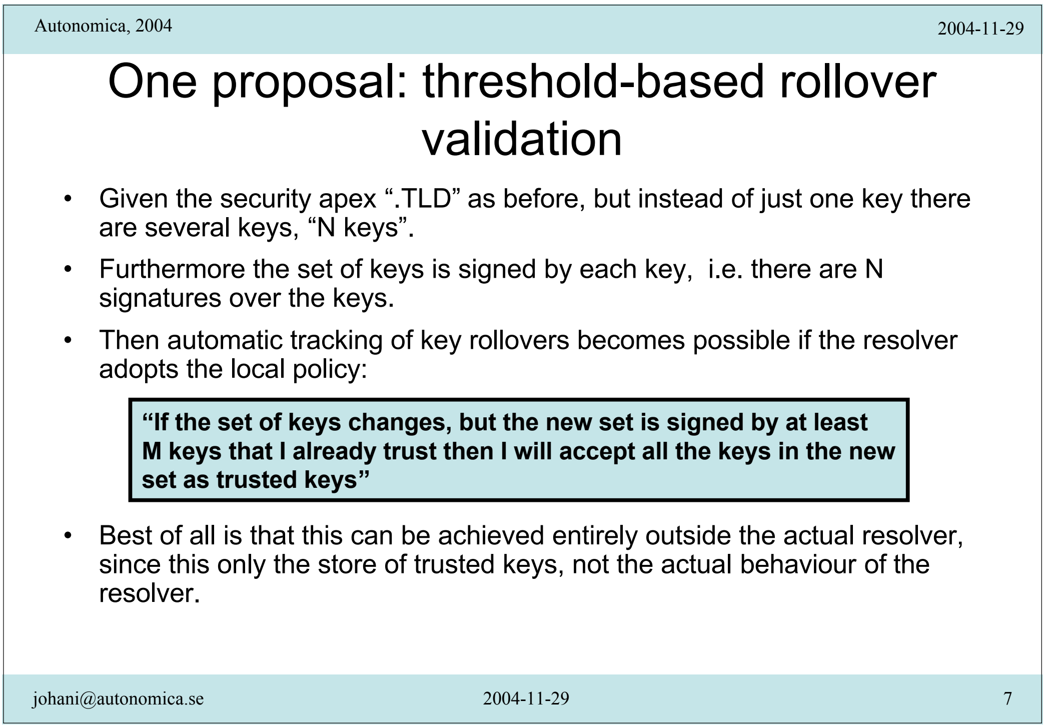
So let’s minimize this cost

- Assume a secure island “.TLD”. This node has at least one DNSSEC key associated with it. This key needs to be replaced over time (this is known as “key rollover”).
- Assume a validating resolver that has this key configured as a “trusted key” to be able to validate lookups.
  - if the resolver fail to notice that the key is rolled validation will stop working. This is bad, and must be avoided.
  - manual tracking of rollovers would be a lot of work
  - also note that especially initially there will be lot’s of secure islands, since not all of the tree will get signed at once



Minimizing resolver costs, cont’d

- There are at least three different proposals for how to improve the situation.
- Note that the decision of what key to trust (i.e. what key will the resolver decide to use as a “trusted-key”) is a local policy decision
  - therefore the mechanisms that aim to minimize the effort needed do not need to affect the actual **protocol**
  - therefore there is no delay in getting the DNSSEC RFCs done associated with this
  - this can be done purely in “policy space” as opposed to “protocol space”

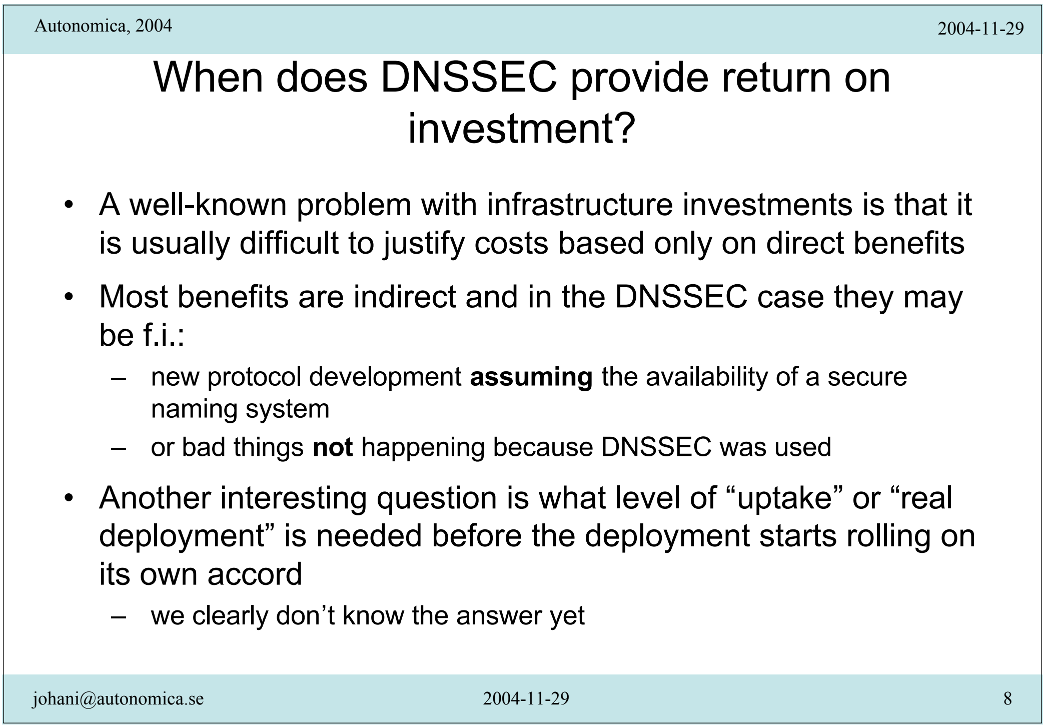


One proposal: threshold-based rollover validation

- Given the security apex ".TLD" as before, but instead of just one key there are several keys, "N keys".
- Furthermore the set of keys is signed by each key, i.e. there are N signatures over the keys.
- Then automatic tracking of key rollovers becomes possible if the resolver adopts the local policy:

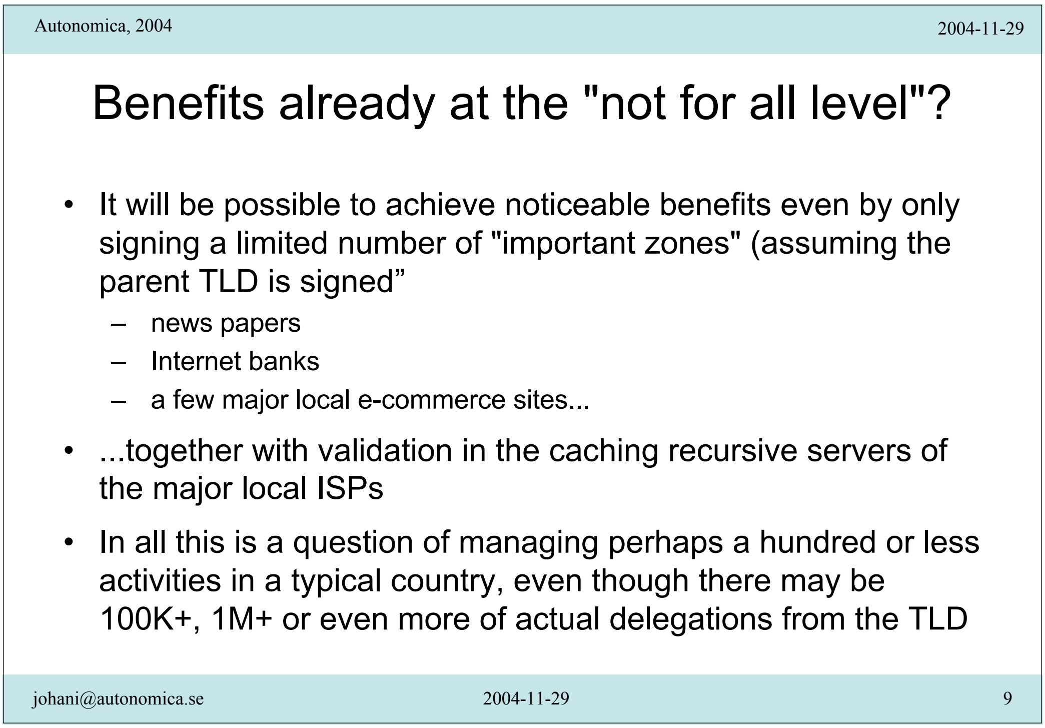
> **"If the set of keys changes, but the new set is signed by at least M keys that I already trust then I will accept all the keys in the new set as trusted keys"**

- Best of all is that this can be achieved entirely outside the actual resolver, since this only the store of trusted keys, not the actual behaviour of the resolver.



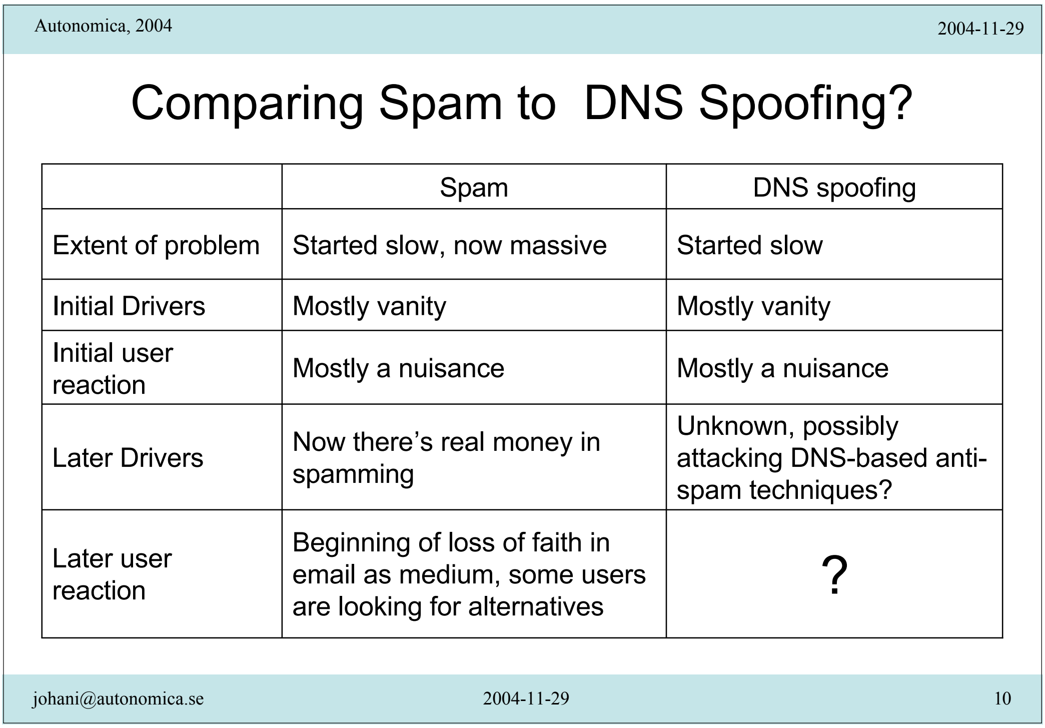
When does DNSSEC provide return on investment?

- A well-known problem with infrastructure investments is that it is usually difficult to justify costs based only on direct benefits
- Most benefits are indirect and in the DNSSEC case they may be f.i.:
  - new protocol development **assuming** the availability of a secure naming system
  - or bad things **not** happening because DNSSEC was used
- Another interesting question is what level of "uptake" or "real deployment" is needed before the deployment starts rolling on its own accord
  - we clearly don't know the answer yet



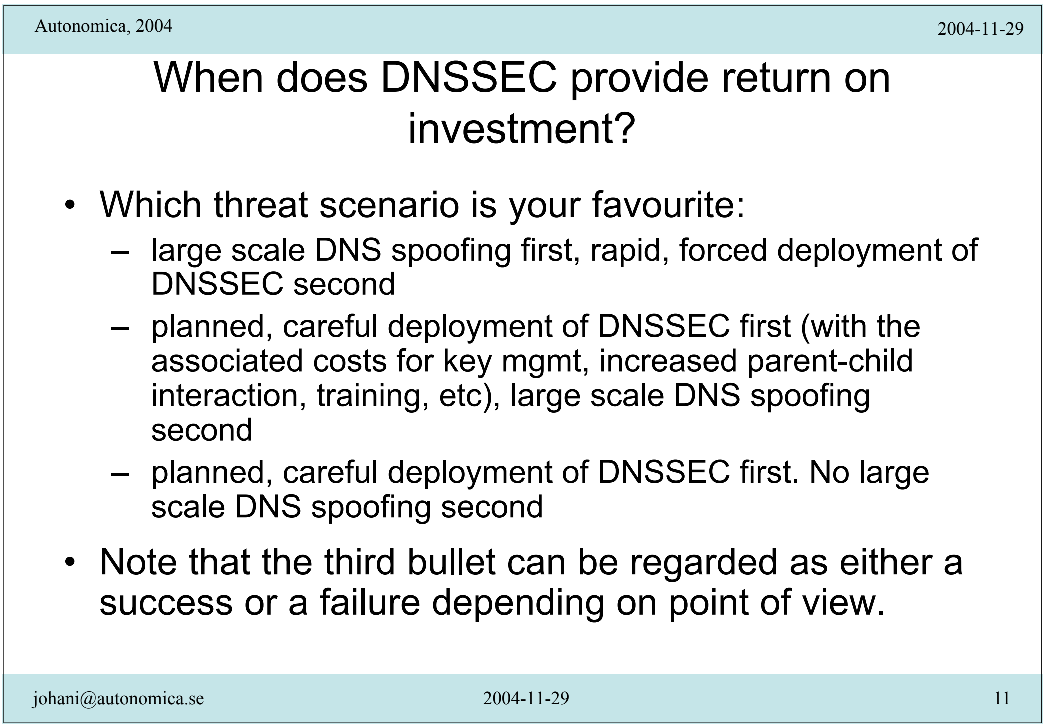
Benefits already at the "not for all level"?

- It will be possible to achieve noticeable benefits even by only signing a limited number of "important zones" (assuming the parent TLD is signed”
  - news papers
  - Internet banks
  - a few major local e-commerce sites...
- ...together with validation in the caching recursive servers of the major local ISPs
- In all this is a question of managing perhaps a hundred or less activities in a typical country, even though there may be 100K+, 1M+ or even more of actual delegations from the TLD



Comparing Spam to DNS Spoofing?

	Spam	DNS spoofing
Extent of problem	Started slow, now massive	Started slow
Initial Drivers	Mostly vanity	Mostly vanity
Initial user reaction	Mostly a nuisance	Mostly a nuisance
Later Drivers	Now there’s real money in spamming	Unknown, possibly attacking DNS-based anti-spam techniques?
Later user reaction	Beginning of loss of faith in email as medium, some users are looking for alternatives	?



When does DNSSEC provide return on investment?

- Which threat scenario is your favourite:
  - large scale DNS spoofing first, rapid, forced deployment of DNSSEC second
  - planned, careful deployment of DNSSEC first (with the associated costs for key mgmt, increased parent-child interaction, training, etc), large scale DNS spoofing second
  - planned, careful deployment of DNSSEC first. No large scale DNS spoofing second
- Note that the third bullet can be regarded as either a success or a failure depending on point of view.

Thanks!

johani@autonomica.se