ETH zürich

Dynamically Recreatable Key (DRKey) Infrastructure

Juan A. Garcia-Pardo
Research Scientist at Network Security Group, ETH Zürich
11 March 2021, PANRG

Presentation Index

What is DRKey?

- Dynamically Recreatable Key Infrastructure is a protocol for key establishment and exchange.
  - Enables entities to share symmetric cryptography keys for authentication.
  - Assumption: All ASes willing to use DRKey have a Key Server.

- DRKey scales well.
  - Impossible to keep state for millions of end hosts.
    - DRKey hierarchy allows distribution of "parent" keys.
  - Derivations are faster than memory lookups for the same key.
    - DRKey uses fast on-the-fly derivations from a root key.
  - Granularity at the Autonomous System level.
    - ASes can blacklist endhosts or whole other ASes.

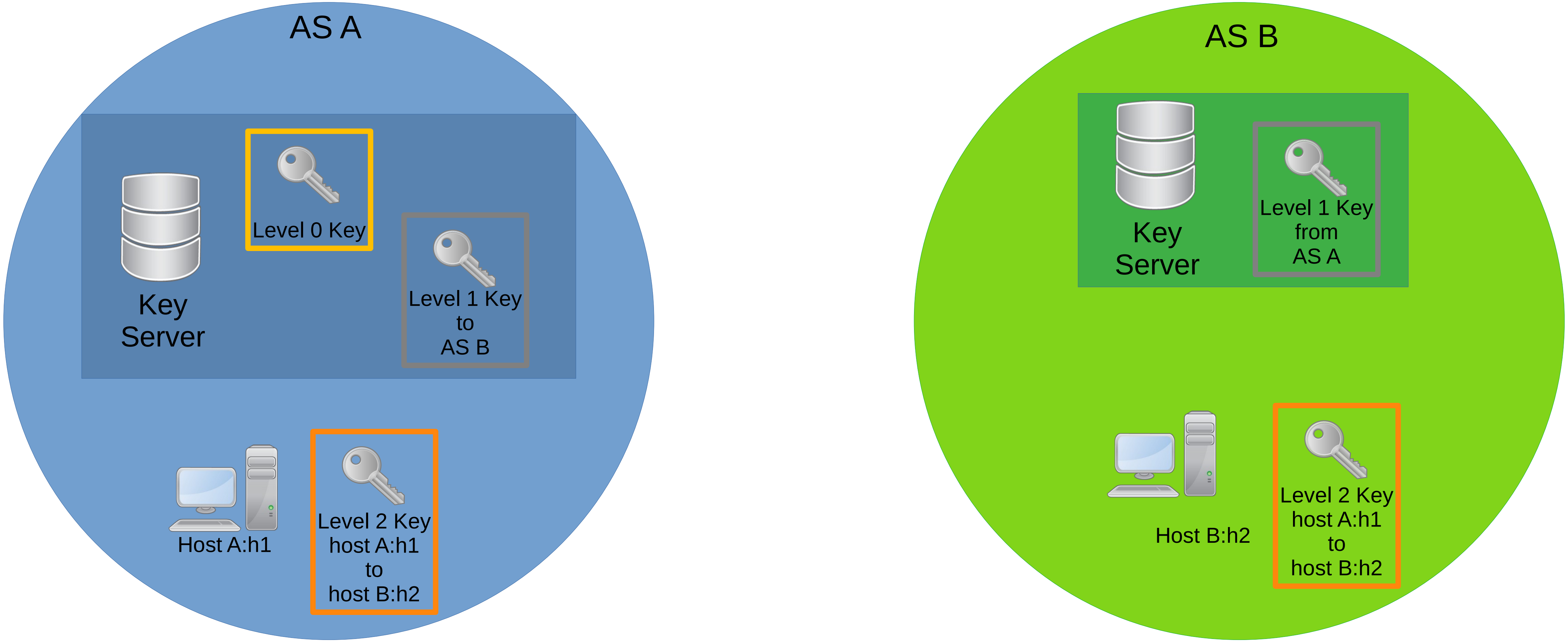
What is DRKey?

- There are three levels of keys:
  - Level 0: the AS secret value. Kept secret in the key server. Used to obtain level 1 keys.
  - Level 1: the AS to AS key. May have locked a protocol (a "purpose"). Used to derive level 2 keys.
  - **Level 2**: the entity to entity protocol key. **Used to authenticate packets.**

- All keys have a validity period. It is established when creating the level 0 key.

- The symmetric cryptography key is obtained in one of two ways:
  - Shareholders of the key: can derive it in nanoseconds.
  - Others: obtain the key via their key server (much slower).

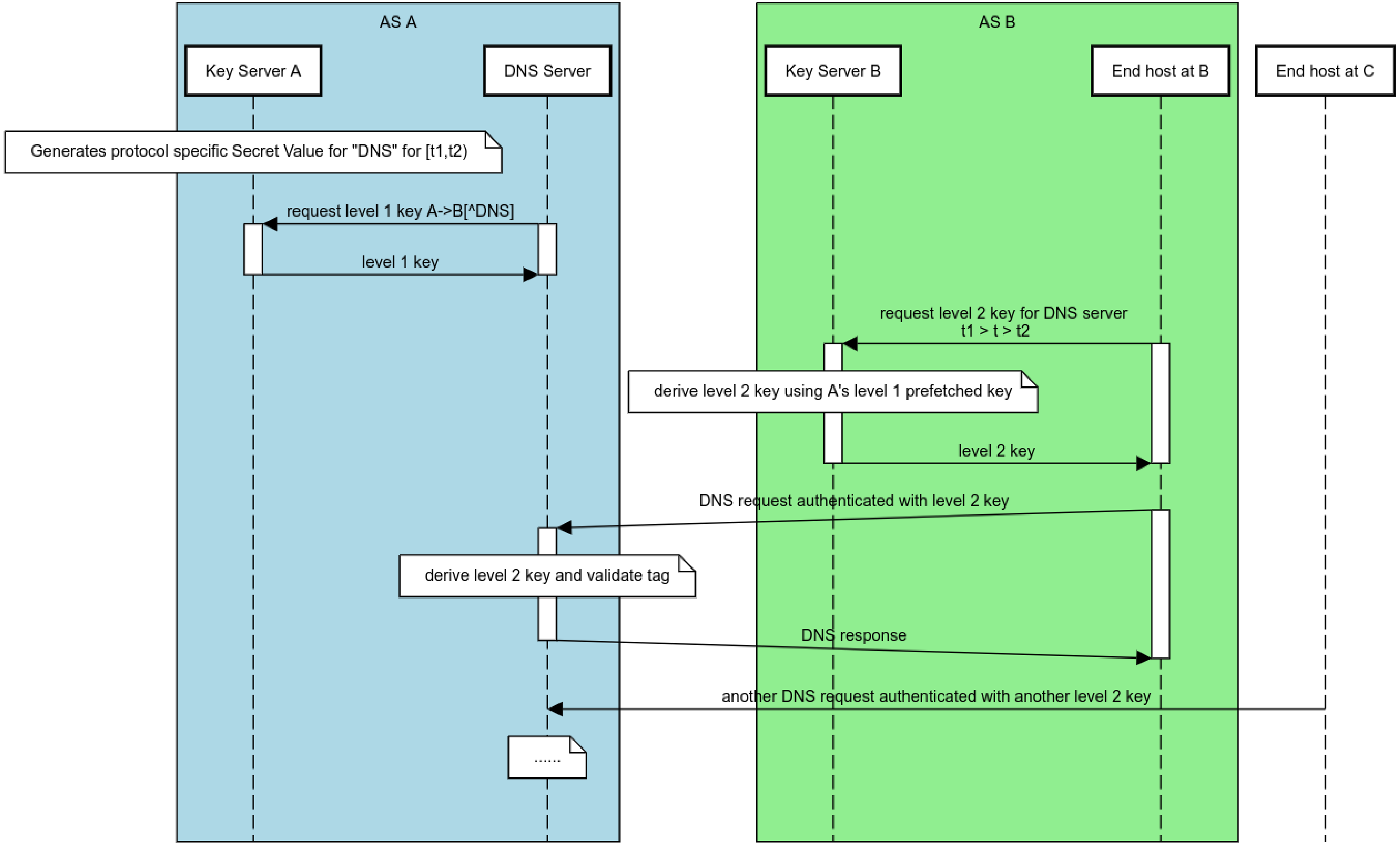
What is DRKey?



Example of Use

- DNS service is authenticating every packet part of a request. Located inside AS A.
  - Servers are busy, they need to derive the key very fast.
- Servers will be trusted by DRKey for certain protocol (we will call it e.g. "DNS“).
- Sequence of events:
  1) Key server in AS A has already the secret value. Derives a secret value for "DNS".
  2) DNS servers in AS A obtain the up to date secret value for "DNS".
  3) End hosts querying a DNS server will first obtain the level 2 key for it. They MAC the packet with it.
     1) Their key servers will first obtain the level 1 key, if not prefetched.
  4) Packet arrives to DNS server. The server extracts the AS ID and endhost IP from metadata.
  5) The server can quickly derive the level 2 key from the metadata and its secret value for "DNS".
  6) The packet MAC is recomputed with this key and checked against the packet‘s tag.

Example of Use



Key Hierarchy

Level 0

- Called Secret Value. Obtained from a master secret and validity period.
  - The only secret used to derive all other keys.
  - Kept inside the key server. Recreated for each validity period (e.g. 24 hours).

Level 1
(to another AS)

- Obtained from a level 0 key and a destination AS.
  - May include a protocol (a string locking its use for a specific purpose, e.g. DNS).
  - Exchanged with other key servers.

Level 2
(from one entity to another entity for a protocol)

- Workhorse key. Used to authenticate packets.
  - Derived from level 1 key. Must contain a protocol.
  - Key servers always derive it. End hosts derive it or obtain it from their key server.
  - Can be used between two hosts, or the shareholder AS and a host or another AS.

Key Hierarchy

- The *levels 1 and 2 keys* are used always between two parties.
  - There is a shareholder side (aka fast side) and the rest (aka slow side).
  - The fast side can derive the key within tens of nanoseconds in software on x86/ARM CPUs. The slow side obtains it from the key server.

- The *level 2 key* is needed to authenticate packets.
  - The goal for the two parties is to have the *level 2 key* when they communicate.
  - Slow side will have to request it beforehand.

Key Derivation Details

- Every derivation is deterministic.
  - Level 0 keys (secret values) use 1000 iterations of SHA256 with PBKDF2 applied to an AS‘ master secret and the validity of the key.
  - Level 1 and 2 keys use AES-CMAC as PRF, keyed on the secret value and level 1 key, respectively.
- Level 1 and 2 key derivations are very fast.

- Nomenclature: $K_{X \rightarrow Y}^{proto}$ denotes a DRKey locked on protocol "proto", that has X as fast path (typically a server) and Y as slow path (typically all end hosts).
  - X and Y can be an AS denoted with a capital letter (e.g. A), or an end host (e.g. A:h1)

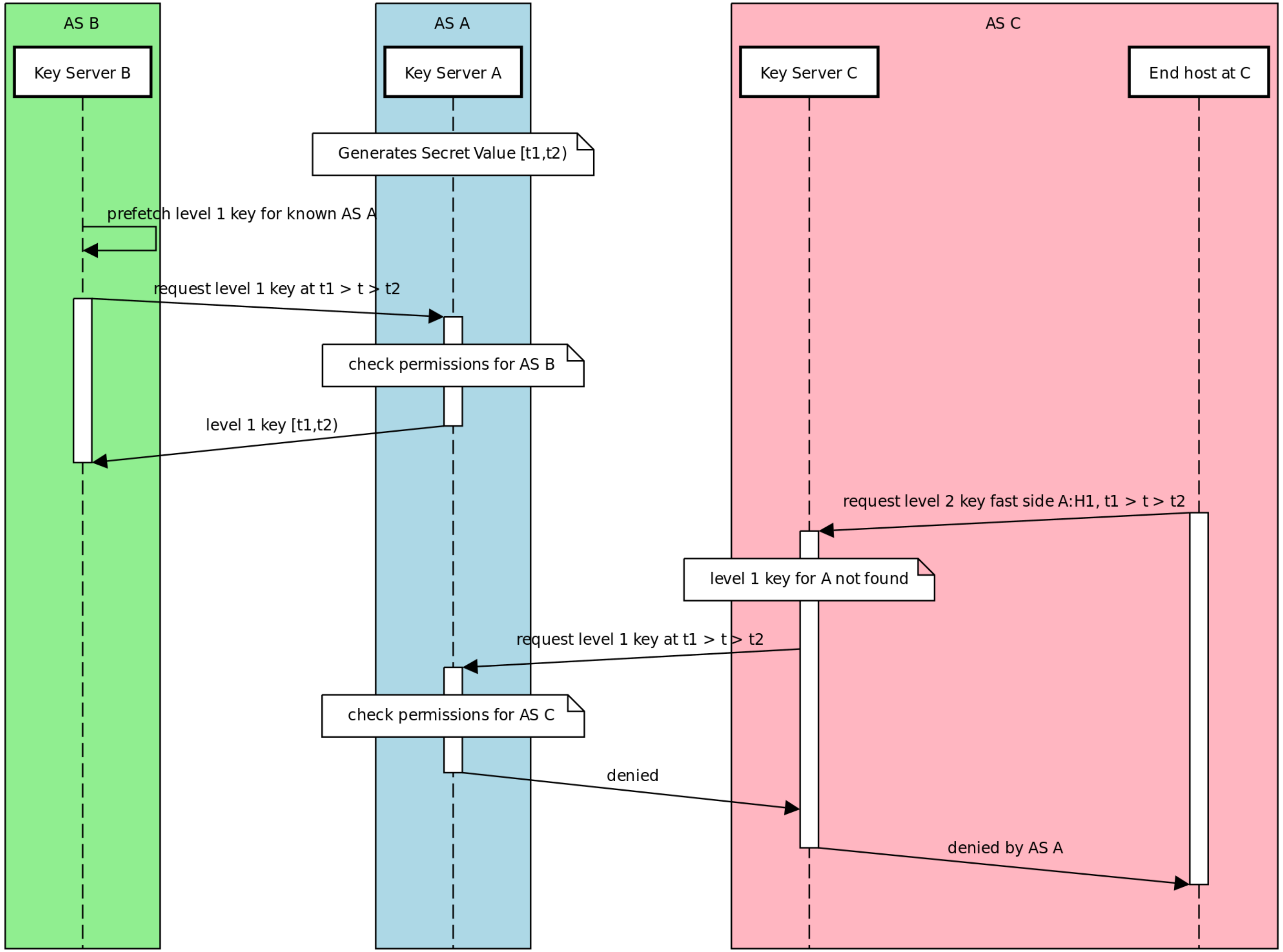
Key Exchange Details

- The level 1 keys must be propagated from the origin (fast side) to all key servers where they could be requested.
- Communication between key servers must be signed and encrypted.
  - In SCION the key servers can use the control plane PKI.
  - In current Internet IP, RPKI can be used.
- Key servers can request level 1 keys to other key servers. These requests can be served or denied, depending on configuration.

- For protocols where the key server could not possibly know the other ASes, the protocol specific secret value must be used, instead of the level 1 protocol locked keys.

- For level 2 keys, key servers could also deny requests, if so configured.

Key Exchange Details



Q&A

- Can DRKey be used for encryption?
- How fast is very fast?
- ...

References

- PISKES paper:
  https://netsec.ethz.ch/publications/papers/piskes_final.pdf

- Netsec Group Webpage:
  https://netsec.ethz.ch/

- DRKey implementation in SCIONLab:
  https://github.com/netsec-ethz/scion/tree/scionlab/go/lib/drkey
  (among other in that repository)

BACKUP SLIDES

(use when time permits / to answer questions)

Key Hierarchy (extra)

- *Level 1 keys* can be locked to specific protocols.
  - This increases security by not exchanging *level 1 keys* "free for all protocols".
- On the other hand, *secret values* can be locked to specific protocols.
  - It allows fast derivation without prior knowledge of the other AS.
- Key servers are trusted. Each AS decides which key servers to trust for which protocols.
  - Other key servers do not have access to the secret value. But they do to the level 1 keys.
  - Key servers typically derive and serve level 2 keys for their end hosts.

Key Derivation Details

$$\text{Secret Value} = SV_A = PBKDF2(\text{validity}, \text{salt}, 1000iter, \text{SHA256})$$

$$\text{Level 1 Key}_{\text{shareholder}=A,\text{other}=B} \equiv K_{A\to B} = \text{PRF}_{SV_A}(B)$$

$$\text{Level 2 Key}^{\text{protocol}} \equiv K^{protocol}_{A:h1\to B:h2} = \text{PRF}_{K_{A\to B}}(\text{"protocol"}, h1, h2)$$

Other possible derivations:

$$\text{Protocol Specific Secret Value} \equiv SV^{proto}_A = \text{PRF}_{SV_A}(\text{"}proto\text{"})$$

$$\text{Protocol Specific Level 1} \equiv \tilde{K}^{proto}_{A\to B} = \text{PRF}_{SV^{proto}_A}(B)$$

$$\text{Protocol Specific Level 2} \equiv \tilde{K}^{proto}_{A:h1\to B:h2} = \text{PRF}_{\tilde{K}^{proto}_{A\to B}}(h1, h2)$$

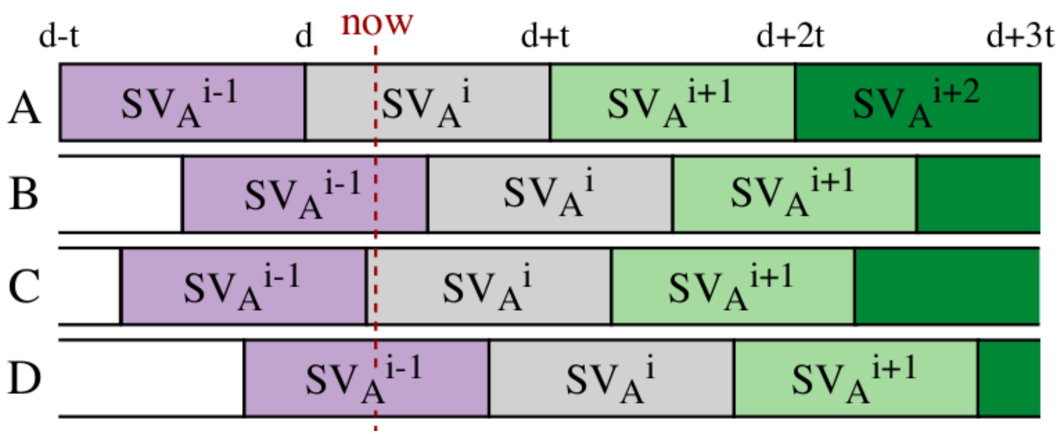
Key Exchange Details

- Because it is typical to have the same validity period (e.g. 24 hours) for many level 1 keys, there could be peaks of level 1 key requests.
- To avoid the concentration, a deterministic function offsetting the validity of the key is used:

$$\text{offset}(A, B) \mapsto [0, t)$$

$$\text{offset}(A, B) = \mathsf{H}(A||B) \mod t$$

- H is a (non cryptographic) hash function.
- The requests are spread uniformly.



Key Server Discovery

- In SCION, the key server can be reached using an anycast address.
- In the current Internet, RPKI can be used (again) for this purpose.
  - E.g. encoding the IP of the key server into a separate extension.