ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

I. Middleboxes No Longer Considered Harmful

II. A Layered Naming Architecture for the Internet

Seminar in Distributed Computing

Louis Woods / 14.11.2007

Intermediaries

- NATs (NAPTs), firewalls and other layer-violating intermediate network elements are collectively known as **middleboxes**.
- Middleboxes violate important **architectural principles** and as a result make the Internet less flexible.
- However, middleboxes exist for important reasons: security (firewalls), private address realms (NATs), performance (load balancing, caching).
- **New perspective:** not middleboxes are to blame, but the Internet architecture itself.
- **Objective**: build extension to the current Internet architecture, that not only allows, but facilitates, the deployment of middleboxes and in such retain their functions while eliminating dangerous side-effects.
- A new name for middleboxes: **intermediaries** (middleman).

Two Principles at the Network Layer

Principle #1:

"Every Internet entity has a unique network-layer identifier that allows others to reach it."

Violation of this principle due to: private networks (NAT), host mobility (notebooks) etc.

Principle #2:

"Network elements should not process packets that are not addressed to them."

Only network elements identified by an IP packet's destination field should inspect the packet's higher-layer fields.

Violation of this principle due to: NATs, firewalls, transparent caches etc.

Delegation-Oriented Architecture:DOA

DOA is meant to be an extension to the current Internet architecture and should fulfil the following requirements:

- Intermediaries can be deployed easily and without having to violate principles #1 and/or #2.
- The architecture allows end-system protocols to evolve independently and quickly.

DOA is based on two main ideas:

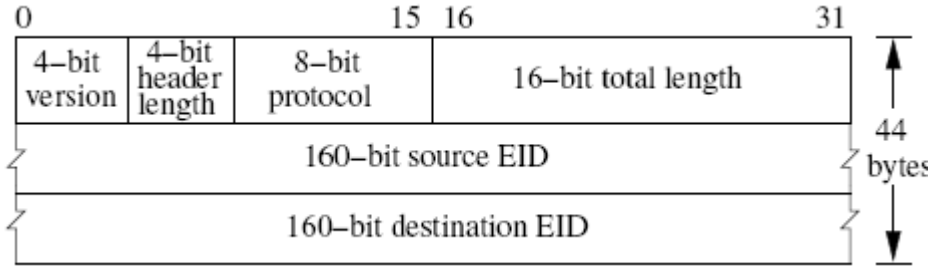
- All entities have a **globally unique identifier**, and packets carry these identifiers.
- **Delegation** as a primitive: DOA allows senders and receivers to express that one or more intermediaries should process packets on the way to a destination.

Endpoint Identifiers (EIDs)

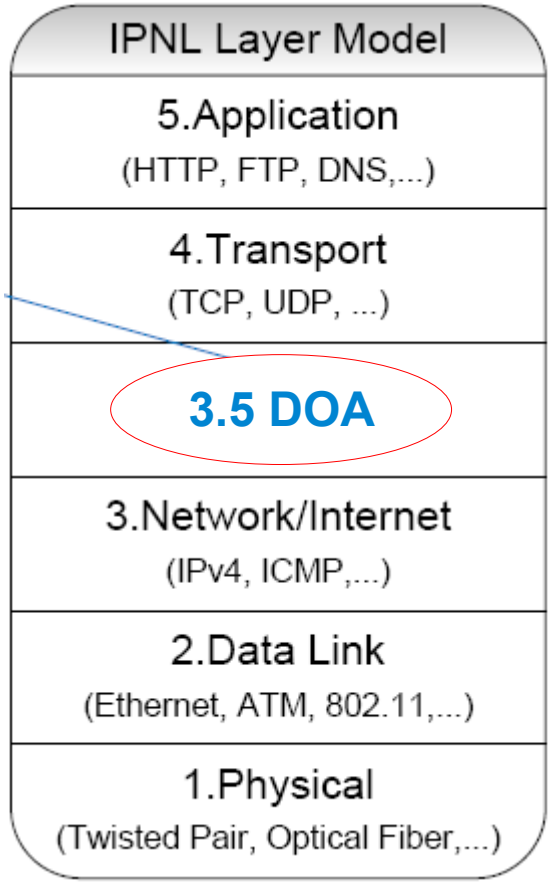
- The identifier should be independent of network topology for hosts to be able to change locations while keeping the same identifier. This rules out IPv6 as a candidate.
- The identifier should be able to carry cryptographic meaning (details later). This means we cannot use human-friendly DNS names either.
- **Solution**: Each host has a globally unique (160-bit) **EID** picked from a large **flat namespace**.
- An identifier namespace is said to be **flat** if the identifiers are unstructured and not overloaded with any semantics about the object being named.
- Example: a **flat identifier** could be a number chosen uniformly at random from an interval such as $[0,2^{160}-1]$.

DOA Header

- Packets are still delivered over IP.
- But transport connections are now bound to source and destination EIDs (instead of IP addresses).
- To carry the EIDs the DOA Header is introduced:



Source: Paper I: Middleboxes No Longer Considered Harmful



Source: Martin Kaufmann's presentation

EID Resolution Infrastructure

- Steps to send a packet from one DOA host to another (without delegation for now):

  1. Obtain an EID **e** (e.g. resolve the recipients DNS name to an EID).

  2. Resolve **e** to an IP address **i**.

  3. Send packet to **i** using the IP protocol.

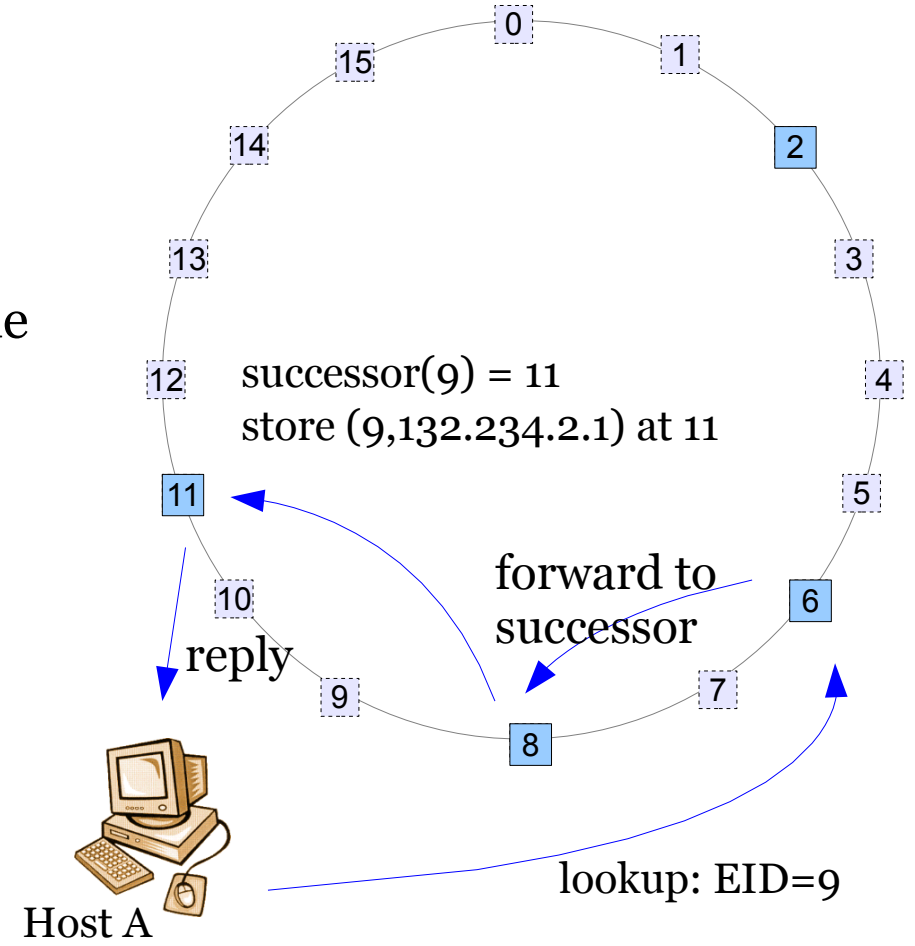
- Resolving a flat identifier like an EID requires a new **resolution infrastructure**. (We can't use anything like DNS as EIDs are not hierarchical).

- Resolution infrastructure must support a **put()/get() interface** over a large, sparse, and flat namespace.

- **Distributed hash tables (DHT)** give exactly this capability.

Resolution of EIDs by Querying DHTs

- Example based on **Chord** (simplified).
- Assume all possible EIDs arranged ordered in a circle [0,2^4-1].
- **successor(e)** = EID of first actual node following e in the circle.
- **Publish**: create tuple(EID,IP) and ask successor(EID) to store that tuple.
- **Search**: Host A sends request to a known node of the resolution infrastructure. Packet is propagated until it locates the successor of the EID which Host A is looking for.



Delegation as a Primitive

- Hosts should be able to express to others to reach the host, packets should be sent to an **intermediary** or a **series of intermediaries**.
- With **delegation** DOA embraces intermediaries as **first-class citizens** which are explicitly invoked and need not be physically interposed in front of the host.
- How does it work? A host can let his EID be resolved to the IP address of a **delegate** (e.g. An intermediary).
- More generally: An EID can also be resolved to a different EID (a delegates **identity**).
- Last but not least an EID can be resolved to a **sequence of EIDs** (where each EID identifies an intermediary specified by the host).

EID-to-IP Mappgins

- Resolve an EID **e** by querying resolution infrastructure (DHT).
- Retrieve an **erecord** from the resolution infrastructure (similar to a DNS resource record).
- Example **erecord**:

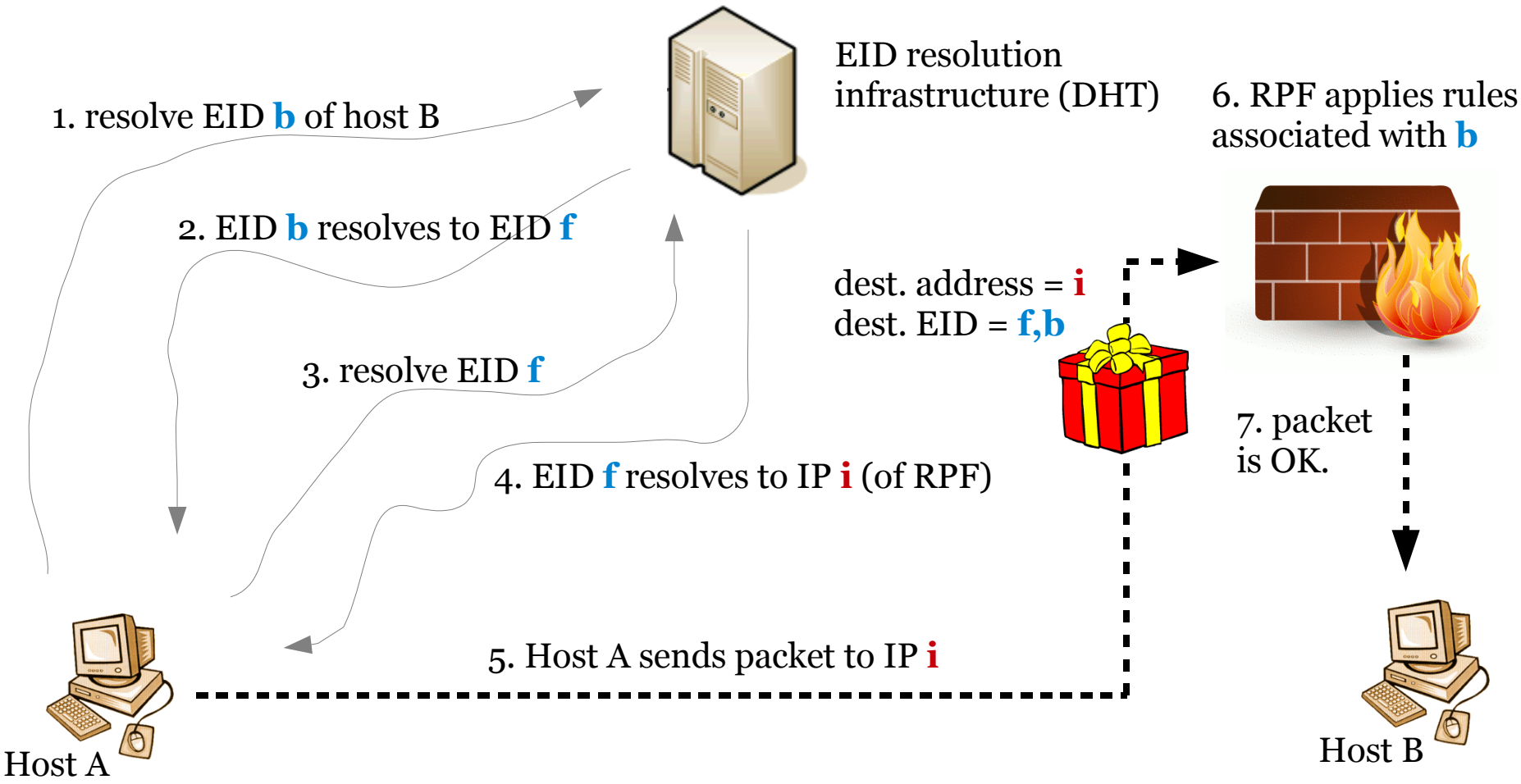
EID: 0x345ba4d ...
Target: IP addr. or EID+
Hint: e.g. IP addr.
TTL: time-to-live (cache)

- **Target** contains only an IP address **i**.
- Send packet to **i**:
  - destination IP = **i**
  - destination EID = **e**
- Semantics: “to reach me, send your packet there”.

EID-to-EID+ Mappgins

- **Target** contains one ore more EIDs **e1**, **e2** …
- Resolve the first EID **e1** in the series to an IP address **i1**.
- In case of **intermediate resolution** to other series of EIDs, insert those EIDs into the original series in logical order (e.g. **e1.a**, **e1.b**, **e2** …).
- Send packet to **i1**.
- The series of EIDs is inserted into DOA header as a stack of EIDs.
- Transport connections are bound to the last EID in that stack.
- Semantics: “to reach me send your packets to these intermediaries in sequence”.

RPF (Remote Packet Filter)



What Happens Exactly at the RPF?

- The RPF extracts the destination EID **b** (the last one in the stack) and looks up the filtering rules for **b** in a local table.
- If the packet passes the filtering rules a **MAC** (Message Authentication Code) taken over the packet and encrypted with a key shared by the RPF and Host B is inserted into the packet.
- Then the RPF rewrites the packet's destination IP address and sends the packet to Host B.
- When Host B gets the packet it recomputes the MAC over the packet and checks if it matches the MAC in the packet.
- Host B ignores packets that fail this test.

Security Issues

- DOA must satisfy the following properties:

  1. Anyone fetching an erecord must be able to verify that the EID owner created it.

  2. Only the owner of an EID may update the corresponding erecord in the DHT.

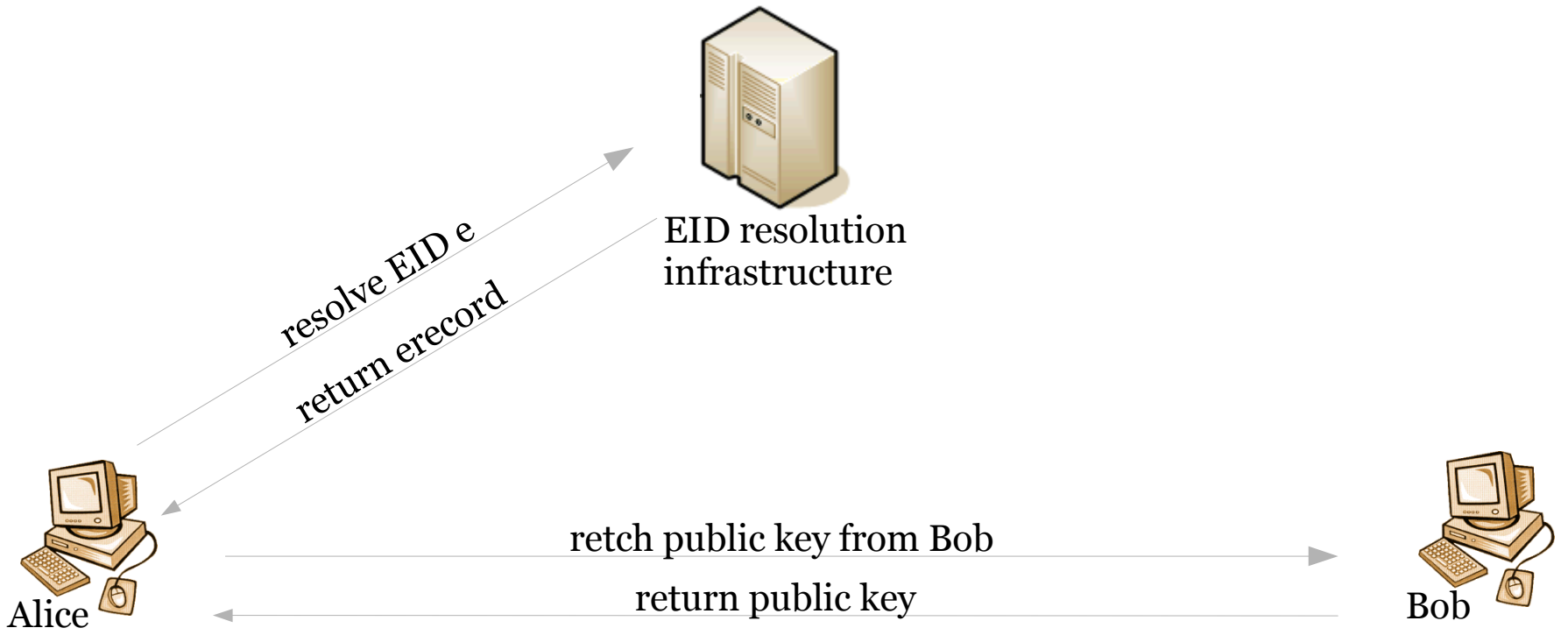
- DOA uses **self-certification** to uphold these properties.

- An EID must be the **hash** of a **public key**.

- The erecord is signed with the corresponding **private key**.

- This is the reason why a requirement for the identifier (EID) was that it should be able to carry **cryptographic meaning** (see earlier slide).

Security (Example) I



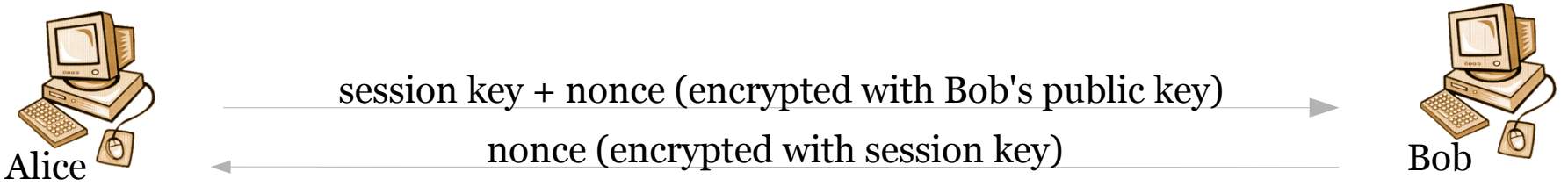
Alice checks **hash(Bob's public key) == EID** ?
Hash function could be for example a SHA-1 function.

Security (Example) II

Alice now knows
she has Bob's public key.

Alice constructs a message containing
a **session key** and a **nonce**.
This message is encrypted with Bob's
public key.

Only Bob has the corresponding
private key. Only he can decrypt the
message and send back the
corresponding nonce encrypted with
the session key.



Alice now knows
she is talking to Bob.

II. A Layered Naming Architecture for the Internet

- Today: Only one level of name resolution (DNS):

  1. domain names → IP addresses

- In this paper they argue there should be **three** levels of name resolution:

  1. **user-level-descriptors** → service identifiers
  2. **service identifiers (SIDs)** → endpoint identifiers
  3. **endpoint identifiers (EIDs)** → IP addresses 

- EID to IP resolution is the exact same mechanism just presented in the previous paper.

Persistent Names for Services & Data

- Today there is no way to directly and **persistently** name **data** and **services**, because naming is always relative to hosts e.g.

  `http://host/dog.jpg`

- Today data and services are treated as **second-class network citizens**.

- **Problems**: when a service or data moves to another host existing references to this service/data will break.

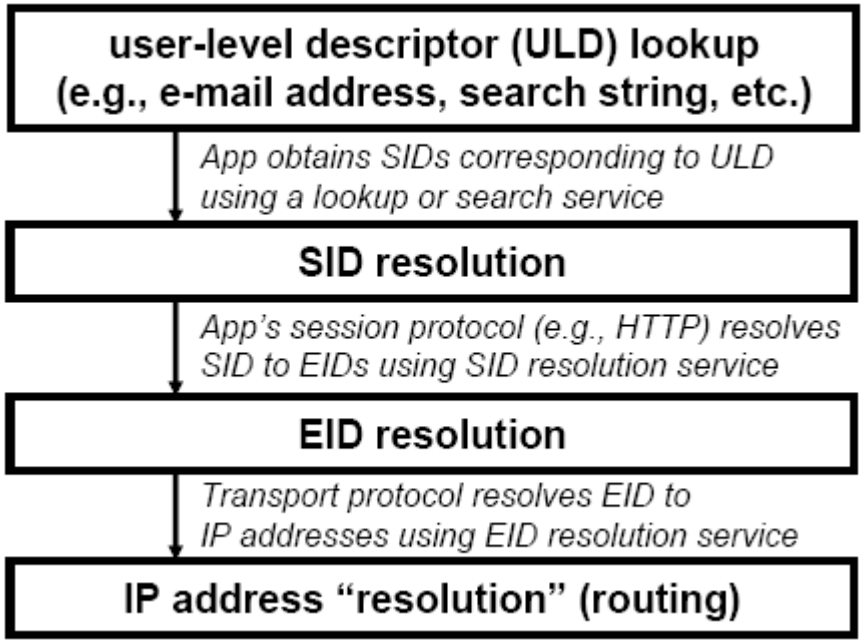
- **Idea**: Separate the service/data names from their hosts.

- How to identify a service? With **service identifiers (SIDs)**. SIDs are persistent names that aren't tied to the endpoint hosting the service.

- As for EIDs also for SIDs a **flat namespace** should be used (no inherent structure and therefore no restrictions on referenced elements).

User-Level-Descriptors

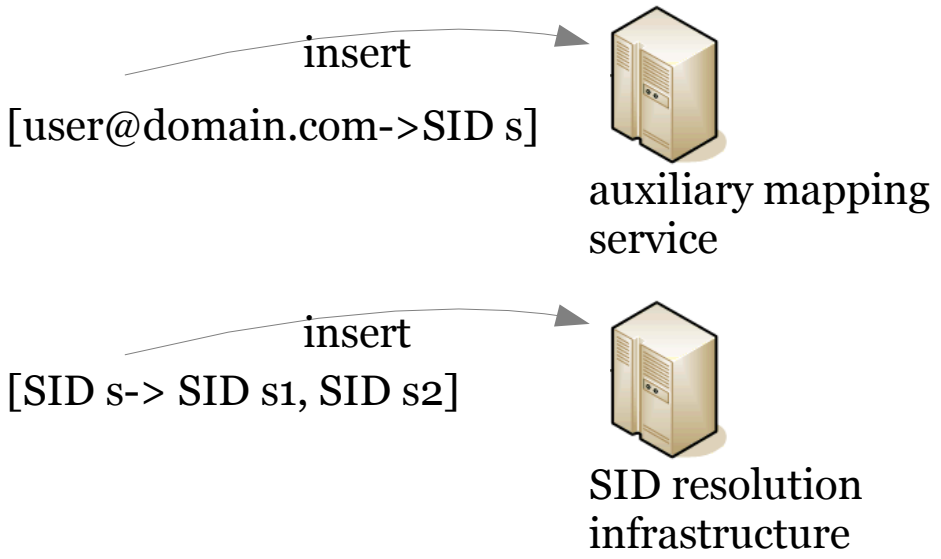
- How does a user obtain an SID? Let's use google.
- A **user-level-descriptor** is a handle in various formats that humans can exchange (e.g. search queries, e-mail addresses).
- So SIDs are the output of different mapping services that take as input **user-level descriptors**.



Source: Paper II: A Layered Naming Architecture for the Internet

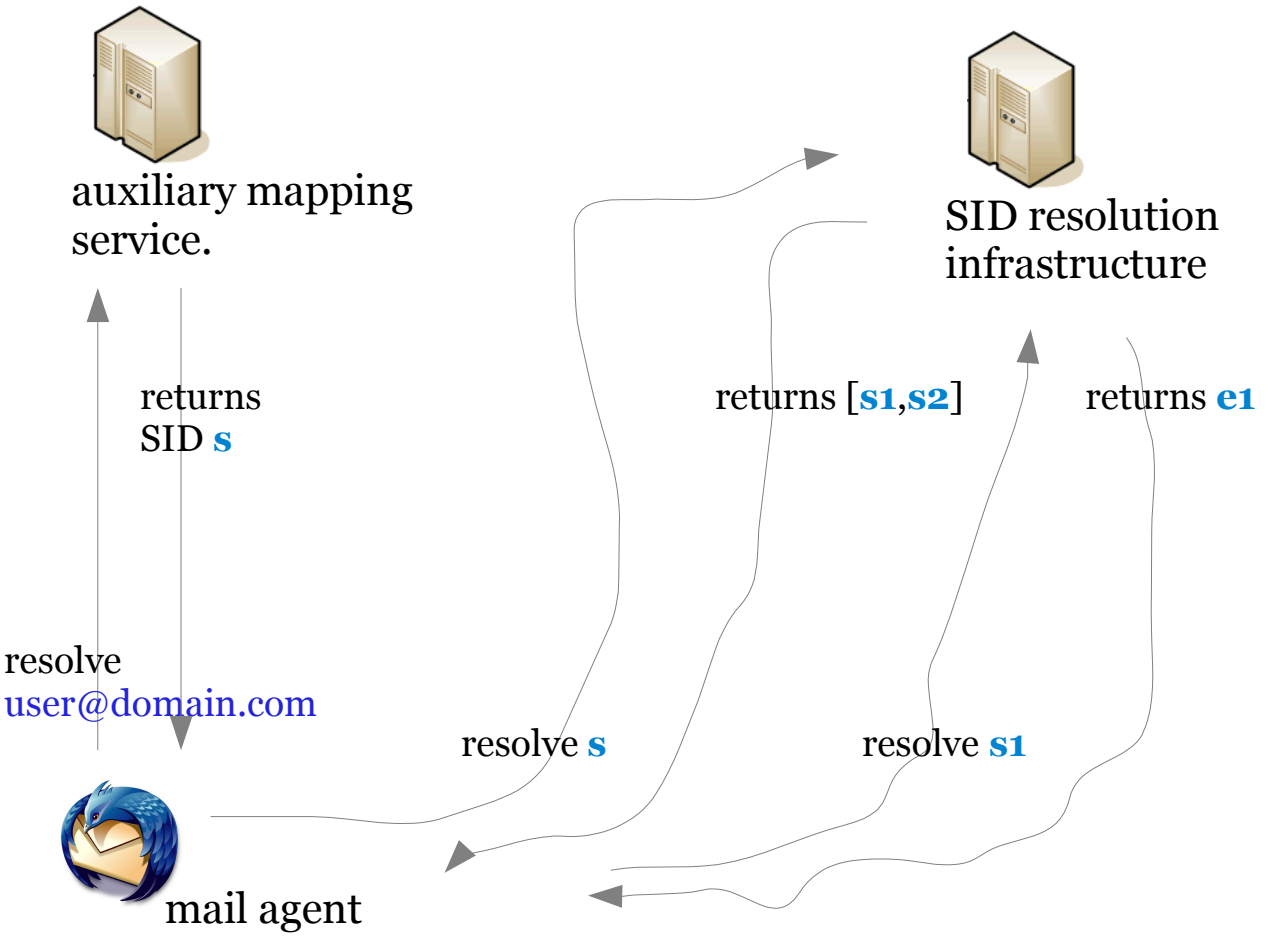
Delegation at the Application Level I

- For example: user@domain.com would like to receive e-mail from an SMTP mail-server after having it first scanned for spam at a third-party site.
- User has to insert the following mappings in the various resolution infrastructures:



user@domain.com : user-level descriptor
s1: identifies third party spam filtering service
s2: identifies the users SMTP server

Delegation at the Application Level II



- Mail agent sends e-mail to spam filter identified by **e1**.
- After e-mail passes the spam-filter the spam-filter resolves **s2** to **e2** identifying the SMTP server and sends the message there.

Things to Remember ...

- **Flat Namespaces**: with DHTs it would be possible to build an extension to the current Internet architecture based on flat names. Flat names can be used to name anything without being restricted by pre-existing structure.
- **Naming Layers**: by separating services and data from endpoints and separating endpoints from network location services, data and hosts can be named persistently yet flexibly.
- **Delegation**: Middleboxes do not need to violate architecture principles. The concept of delegation allows the interposition of intermediaries without violating such architectural principles.

Papers

- **Middleboxes No Longer Considered Harmful**:
  Michael Walfish, Jeremy Stribling, Maxwell Krohn, Hari Balakrishnan, Robert Morris, and Scott Shenker
  Proceedings of USENIX OSDI, San Francisco, CA, ***December 2004***.

- **A Layered Naming Architecture for the Internet**
  Hari Balakrishnan, Karthik Lakshminarayanan, Sylvia Ratnasamy, Scott Shenker, Ion Stoica, and Michael Walfish
  Proceedings of ACM SIGCOMM, Portland, OR, ***September 2004.***

- Both papers can be found here: http://nms.csail.mit.edu/doa/#papers

Questions

Thank you for your attention !