Name-based sockets

A technical overview

Name-based sockets

Presentation overview

- Goals
- A name-based approach
- API
- Name-exchange
- Features made possible – Name-shim6
- Open issues – name exchange & authentication

Goals

Make application development easier

By providing a friendly API
and packaging all that difficult network stuff neatly

Make deployment easier

By using a different deployment strategy

Constraints

Be backwards compatible

Do not require new/changed infrastructure

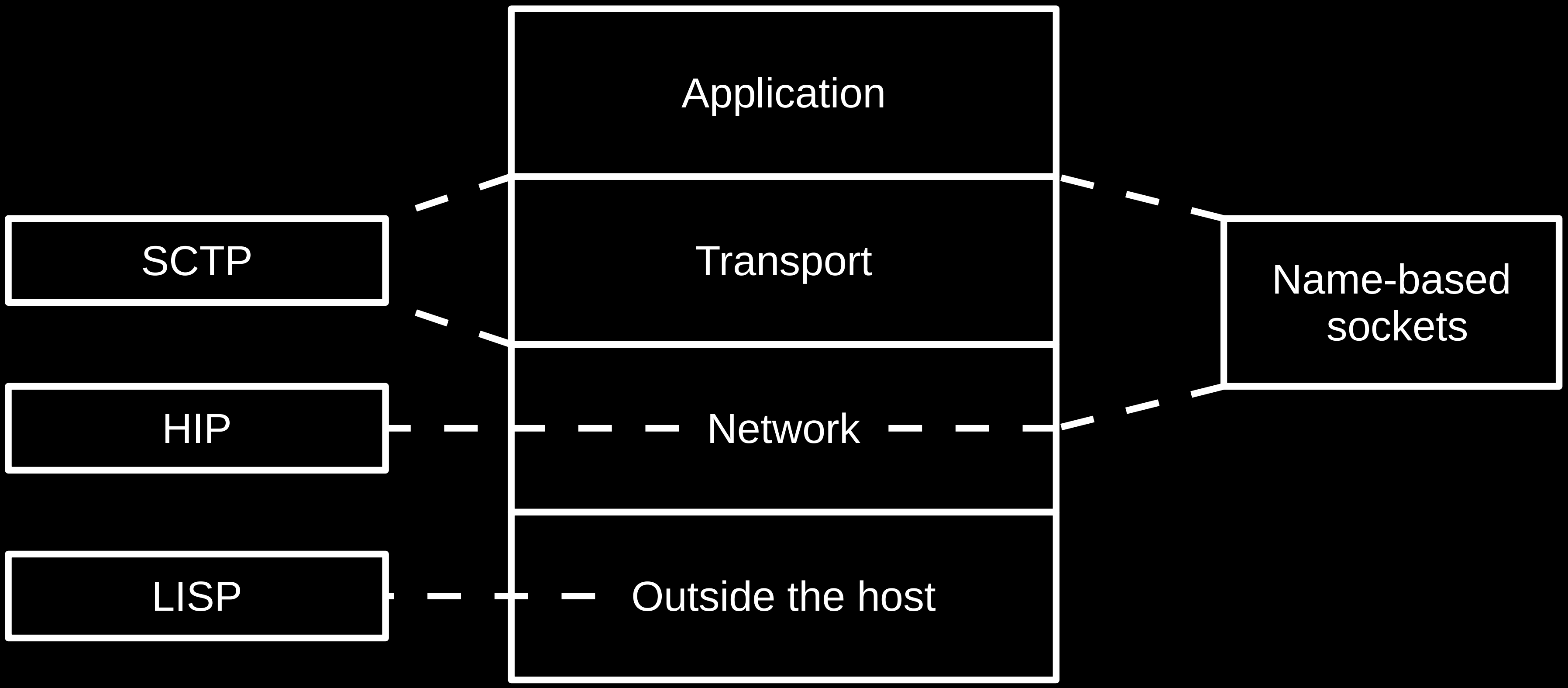
Do not worsen performance

Be configurable

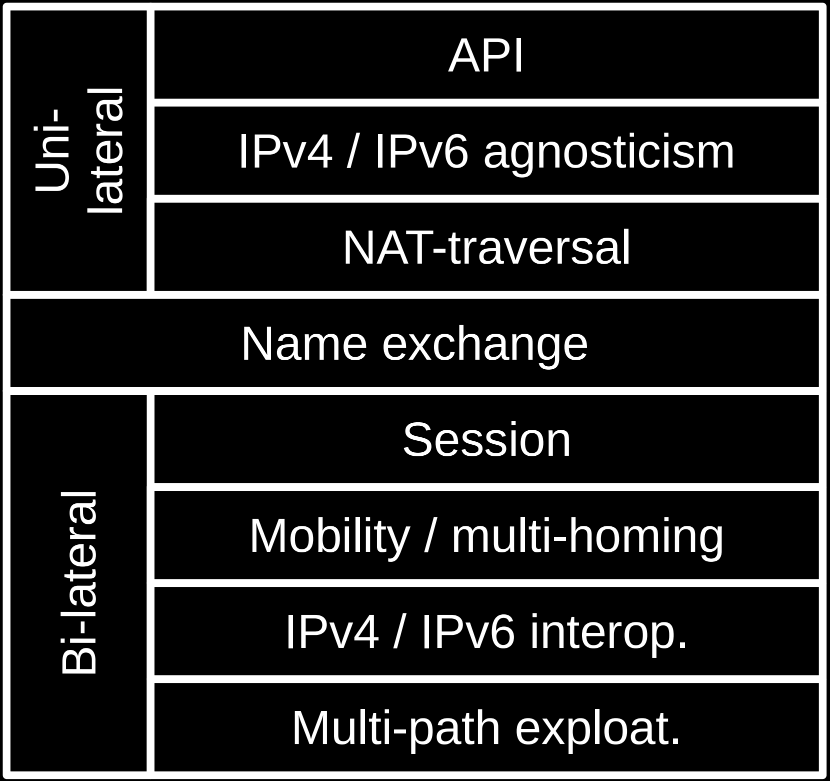
The approach

Host	Network
New API	No changes
A session layer	Compatible
Easy transition	Middle-boxes unaffected
All features are opt-in	

How is this different?



Uni- / Bi-lateral



Uni-lateral

API

IPv4 / IPv6 agnosticism

NAT-traversal

Name exchange

Bi-lateral

Session

Mobility / multi-homing

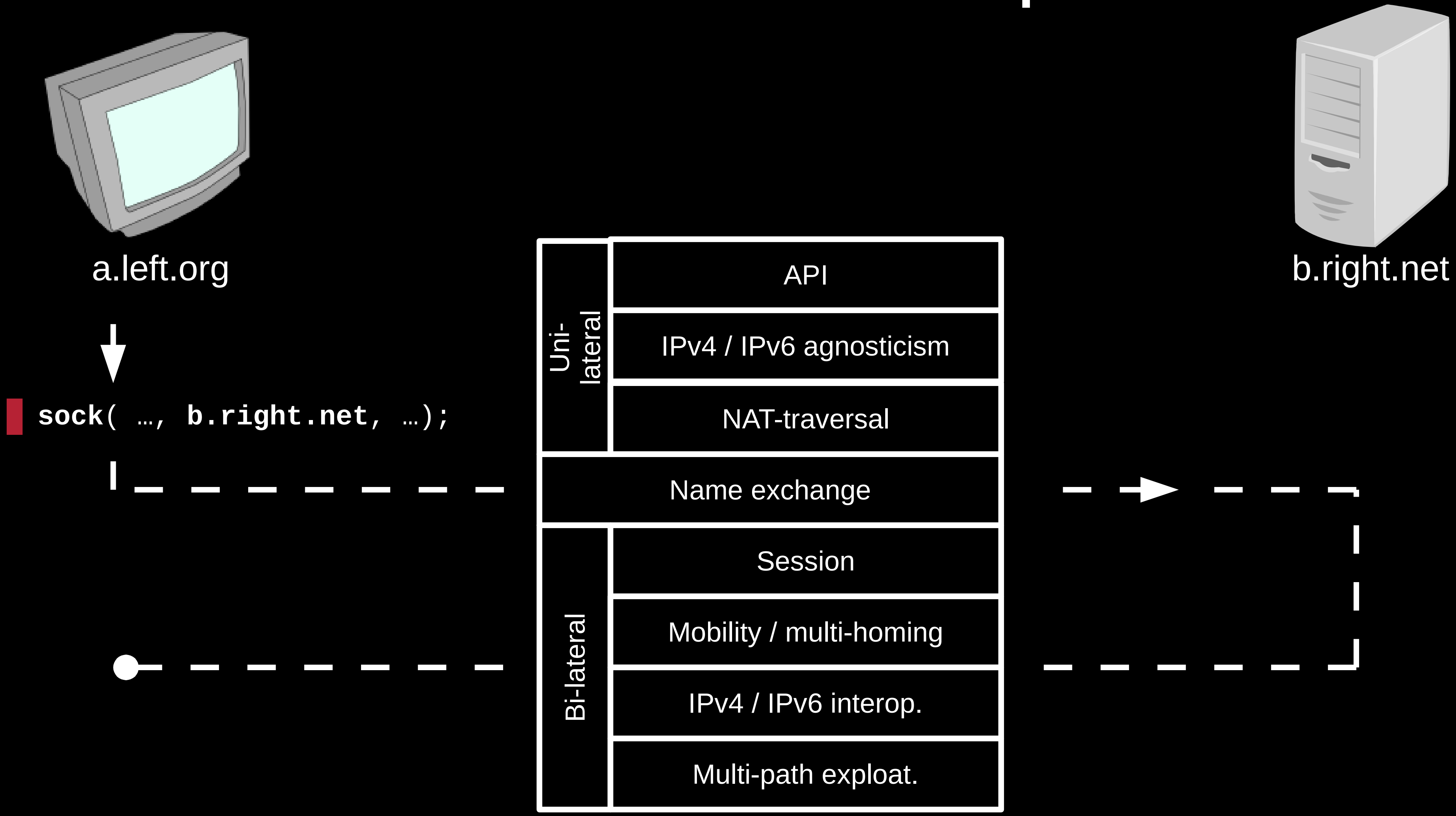
IPv4 / IPv6 interop.

Multi-path exploat.

FQDN

- Not required
- Does help reachability

A time-line example



a.left.org

b.right.net

```
sock( …, b.right.net, …);
```

Uni-lateral

- API
- IPv4 / IPv6 agnosticism
- NAT-traversal

Name exchange

Bi-lateral

- Session
- Mobility / multi-homing
- IPv4 / IPv6 interop.
- Multi-path exploat.

Name-based sockets

Presentation overview (functionality)

- API
  - Current prototype status
- Name-exchange
- Name-shim6
  - A name-based example

API – the vision

```
fd = sock( AF_NAME, dest_name, proto, service);
data = read(fd);
write(fd, data);
```

Use names

Do away with struct sockaddr

Use service names (not ports)

And of course provide some sensible defaults
and a set of setsockopt()s

API – Current status

Names can be provided as either FQDN or as IPs.
IPs are encoded as strings (<ip>.ip6.arpa)

“The IP with which the session started.”

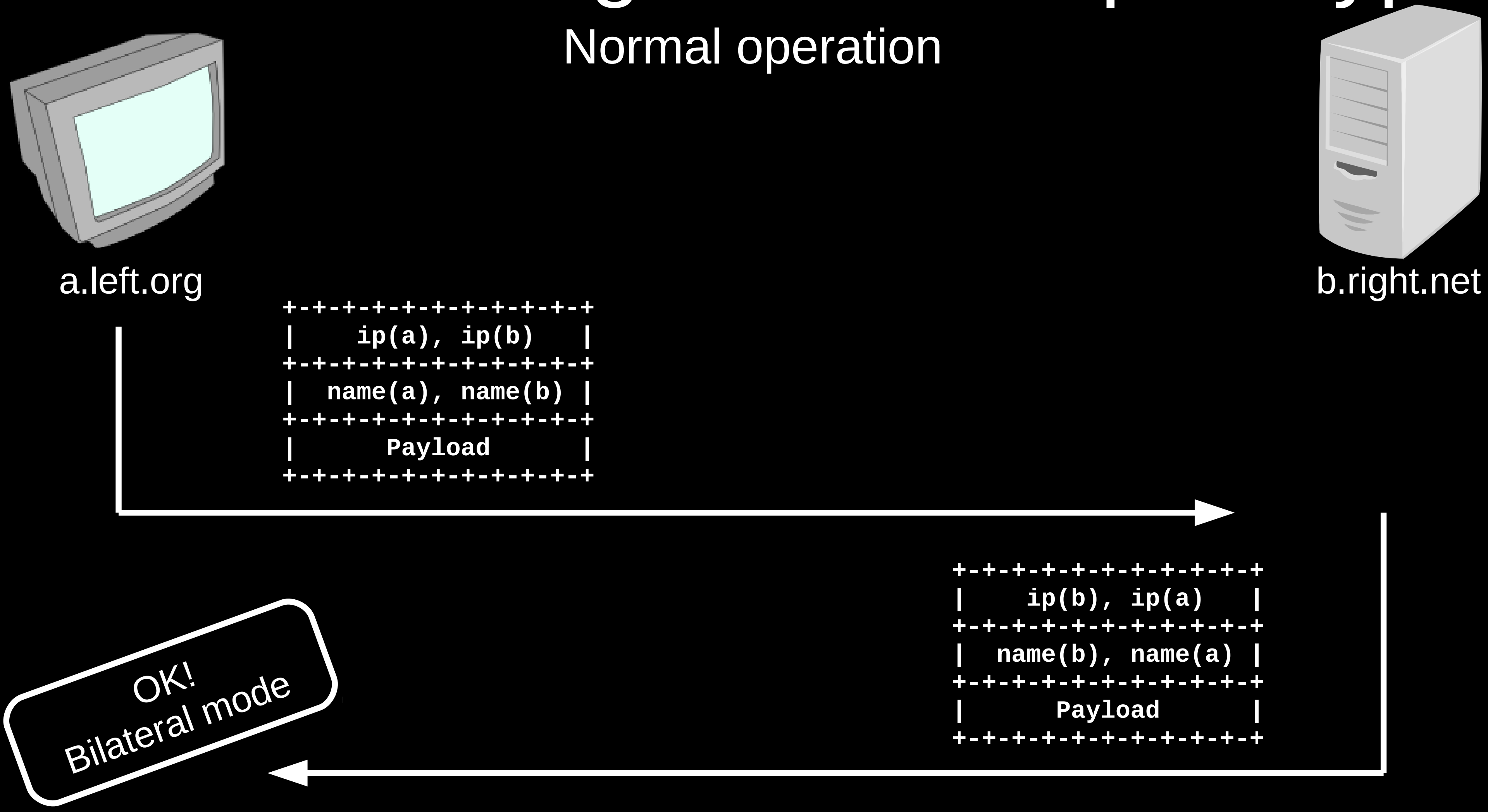
```
fd = sock(AF_NAME, SOCK_STREAM, IPPROTO_IP);

struct sockaddr_name bind_name;
strcpy(bind_name->sname_addr.name, src_name);
bind( fd, (struct sockaddr *)&bind_name,
    sizeof(bind_name));

struct sockaddr_name conn_name;
strcpy(conn_name->name, dst_name);
conn_name.sname_port = htons(port);
connect( fd, (struckt sockaddr *)&conn_name,&len);
```

Name-exchange – current prototype

Normal operation



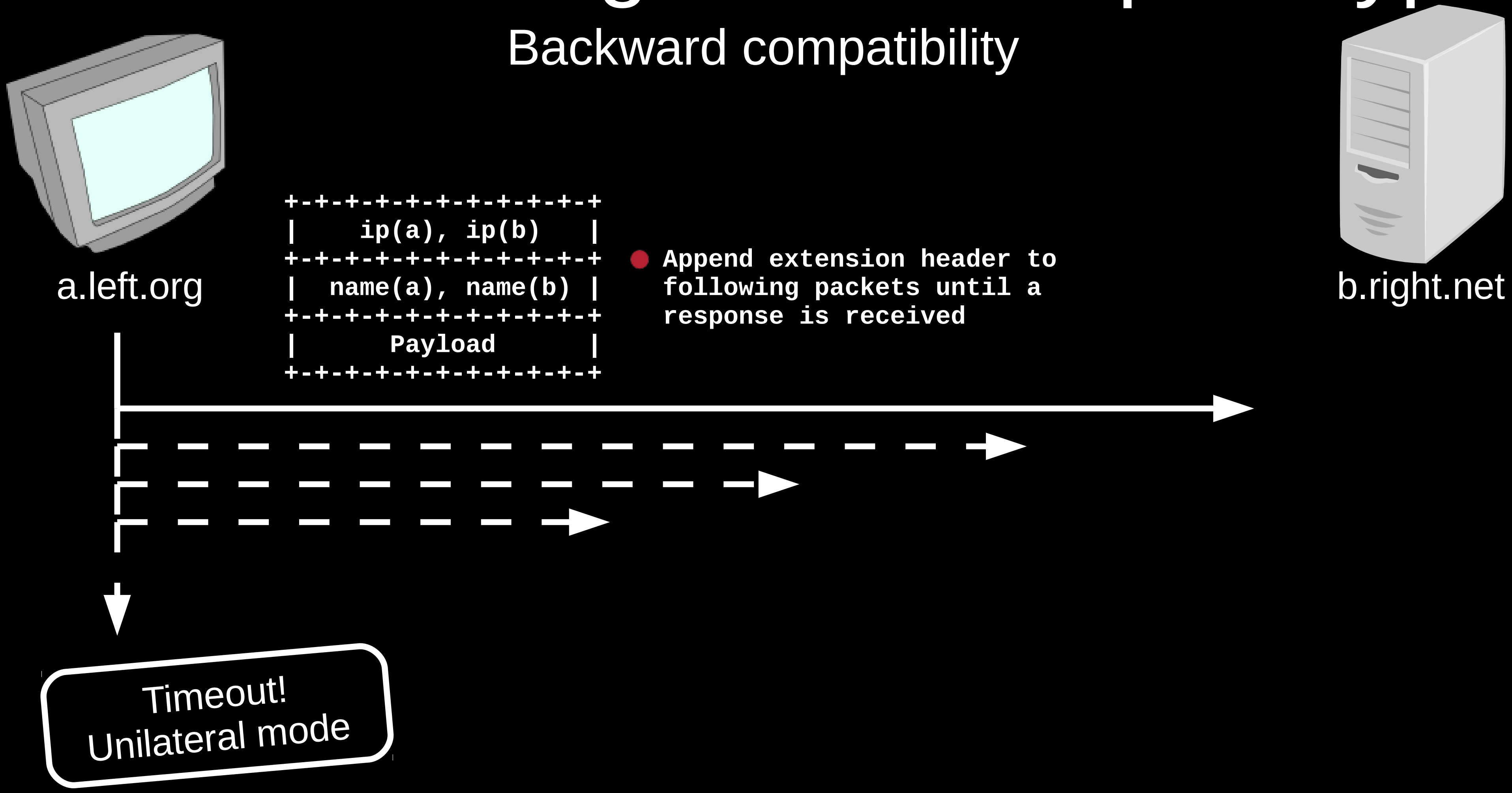
```
+-+-+-+-+-+-+-+-+-+-+-+
|    ip(a), ip(b)    |
+-+-+-+-+-+-+-+-+-+-+-+
|  name(a), name(b)  |
+-+-+-+-+-+-+-+-+-+-+-+
|      Payload       |
+-+-+-+-+-+-+-+-+-+-+-+
```

```
+-+-+-+-+-+-+-+-+-+-+-+
|    ip(b), ip(a)    |
+-+-+-+-+-+-+-+-+-+-+-+
|  name(b), name(a)  |
+-+-+-+-+-+-+-+-+-+-+-+
|      Payload       |
+-+-+-+-+-+-+-+-+-+-+-+
```

OK!
Bilateral mode

Name-exchange – current prototype

Backward compatibility



```
+-+-+-+-+-+-+-+-+-+-+-+
|    ip(a), ip(b)    |
+-+-+-+-+-+-+-+-+-+-+-+
|  name(a), name(b) |
+-+-+-+-+-+-+-+-+-+-+-+
|      Payload       |
+-+-+-+-+-+-+-+-+-+-+-+
```

- Append extension header to following packets until a response is received

Name-shim6 – current prototype

What names can change

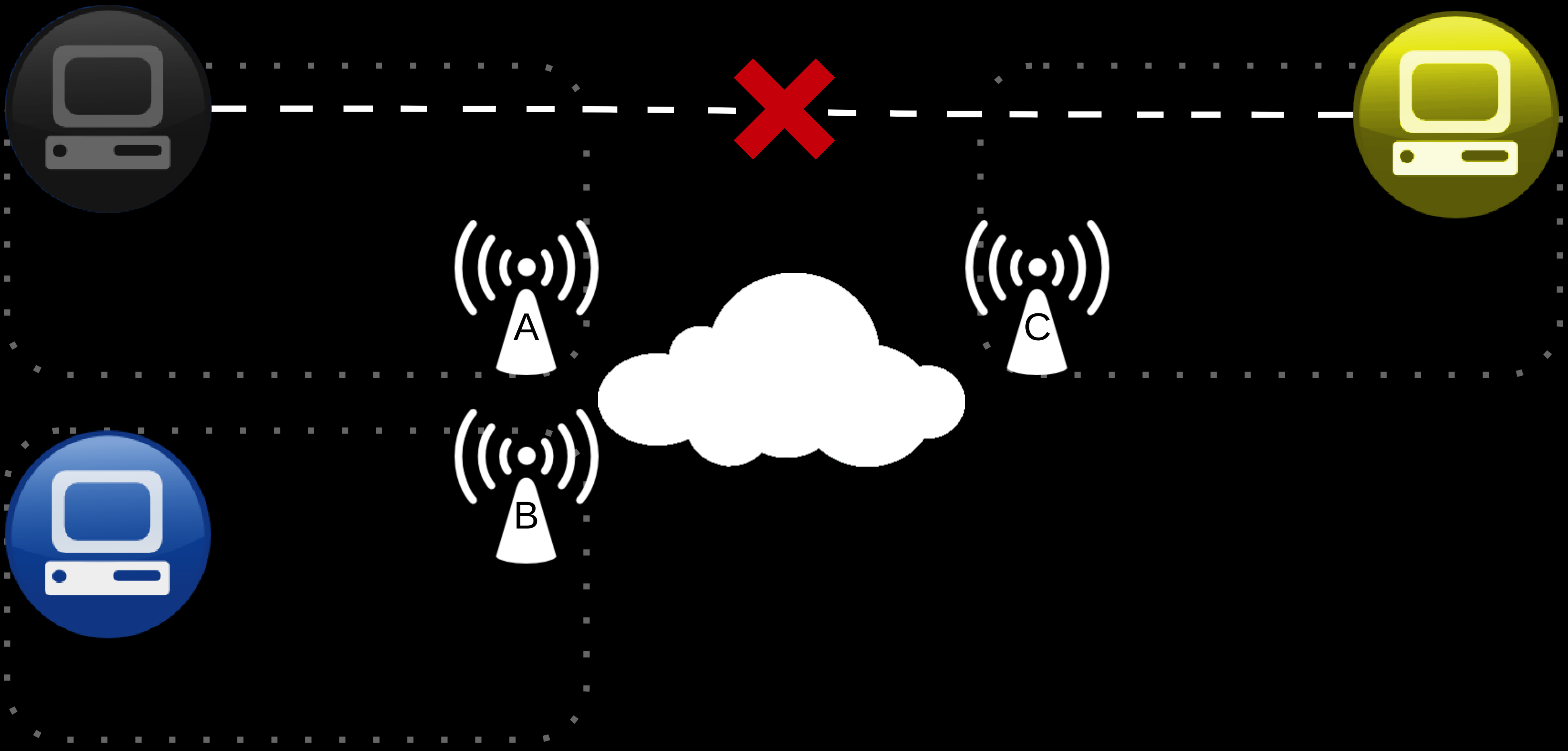
By changing the binding from the upper layers to a name we enable a set of possible enhancements.

We chose to apply a name-based approach to shim6.

Nota bene: Name-shim6 is not a required feature by name-based sockets.
It is an example on which kind of features might be part of a name-based socket and how a name-based approach might benefit existing solutions.

Name-shim6 – current prototype

What names can change



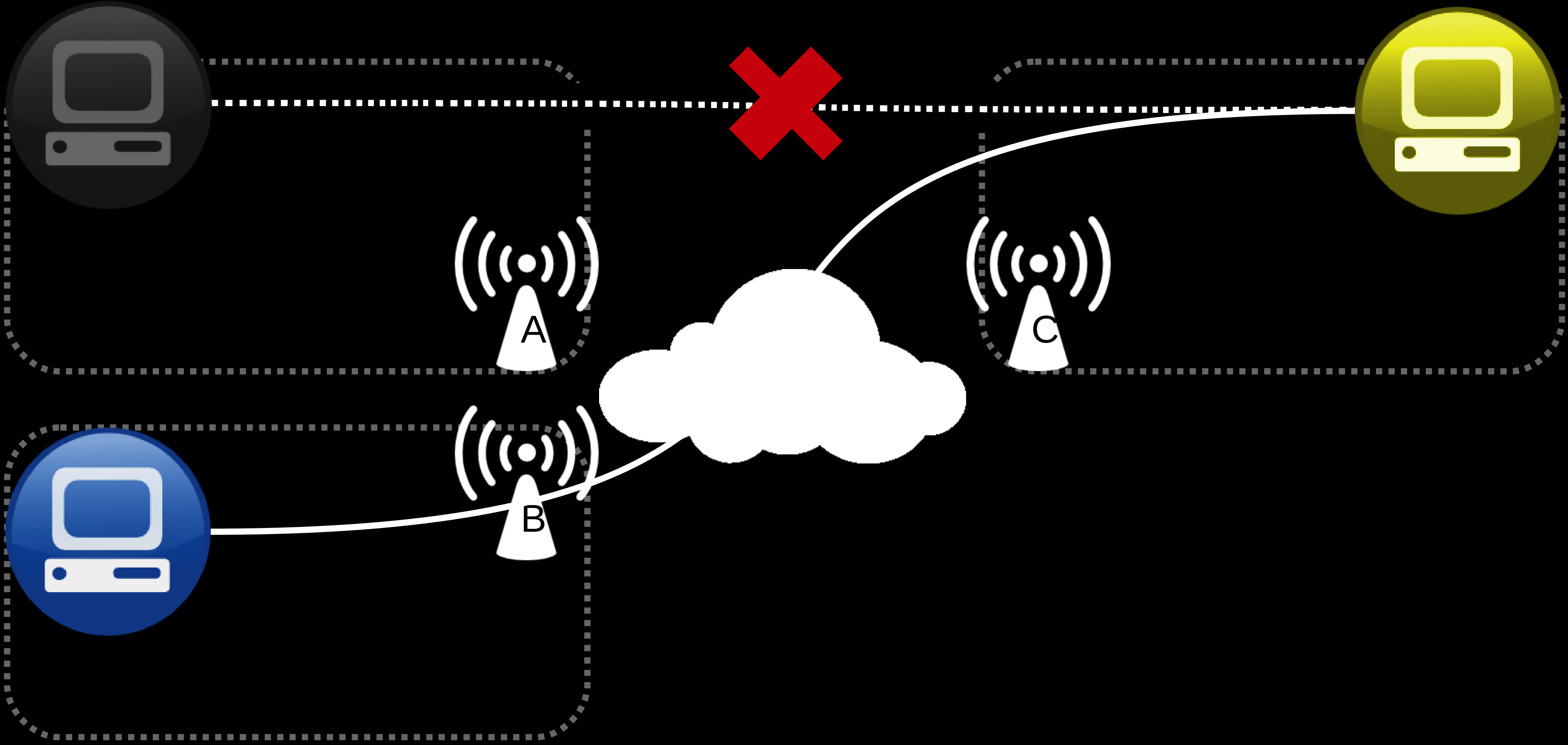
Name-shim6 – current prototype

What names can change

Vanilla shim6	Name-shim6
ULID = Locator	ULID = Name
Out of the hosts control	Host is in control of the validity of the Name.
If this locator becomes Invalid the connection must be shut down	Multi-homing → Mobility

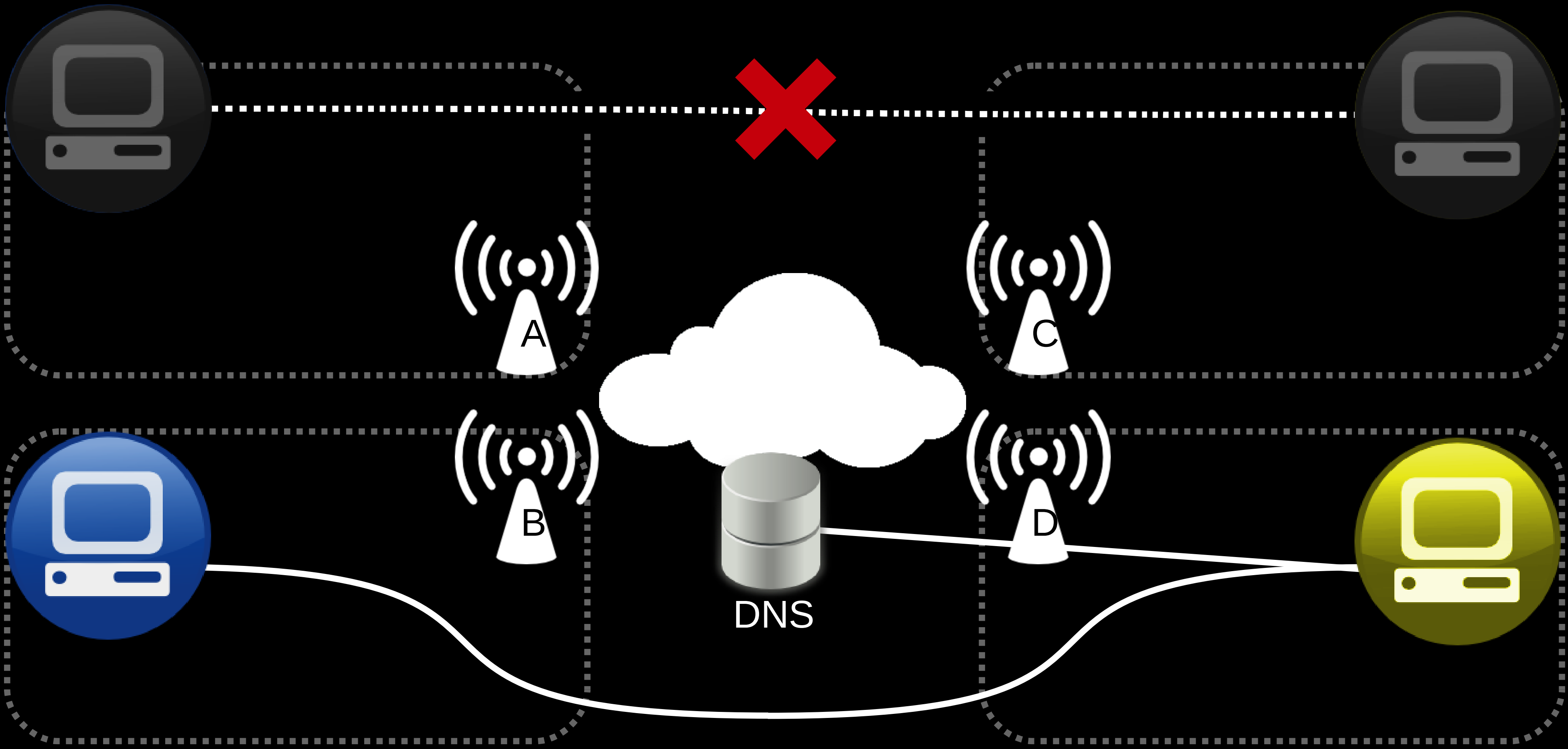
Name-shim6 – current prototype

What names can change



Name-shim6 – current prototype

If (name == FQDN)



Name-shim6 – current prototype

If (name == FQDN)

If the name is resolvable to a locator
the ID → locator substrate can be used
as a rendezvous-point.

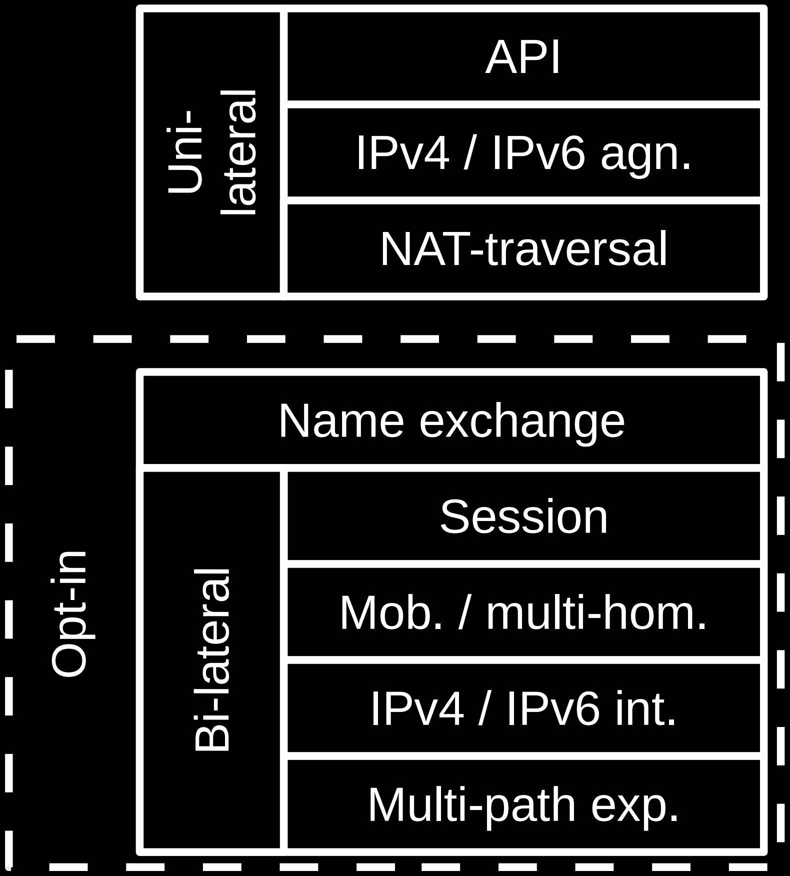
DNS is already out there!

Name-exchange

Opt-in-ness

Name-based sockets functions should all be opt-in.

The name exchange and authentication of the ID → locator binding SHOULD be done if and when requested by a feature.



Name-based sockets

Re-iteration

- API
- Name-exchange
- Name-shim6
- When should the names be exchanged?
- When should name → IP be authenticated?

Name-based sockets

Thank you!