



User-Oriented Networking

Mark Allman, ICSI
Michael Rabinovich, CWRU
Nicholas Weaver, ICSI

FIND PI Meeting
April 2009

*“Well, beat the drum and hold the phone,
the sun came out today!”*

Observations

- Content providers are now normal people
 - picture sharing, blogs, wikis, etc.
- Massive application variety
 - AV, instant messages, commerce, file sharing, social networking, email, gaming, information retrieval, virtual worlds, etc.
- User access network with an increasing number and variety of devices
 - computers, phones, iPods, Kindles, etc.

Implications

- These trends have lead to an immense amount of user-specific meta-information
 - not content, but *how to get content*
 - social graphs, names, application state, etc.
- We have designed ad-hoc coping strategies
 - bookmark sharing, address book merging, IM gatewaying, social graph creation, etc.

Example: Email

- Consider:
 - allman@icsi.berkeley.edu
 - mark@icir.org
- Context-insensitive
- Hard to share
- Ambiguous for people
- Coupled with particular service provider

Example: Email (cont.)

- Coping #1: address books
- Coping #2: institution independent address
- Coping #3: email forwarding services
- Coping #4: Google

A New Primitive

- Rather than point-wise coping strategies, a general mechanism to store user's meta-information
- MISS == Meta-Information Storage System
 - analogous to DNS for user-related data

The End (Part I)

- We propose a *foundational service* that holds users' meta-information

MISS

- Can we build it?
- What does it look like?

Requirements

- System must be *extensible* to arbitrary meta-information
- System must allow *fine-grain access control* to limit proliferation personal meta-information
- System must provide for meta-information *integrity*
- System must allow users' meta-information to be *portable* and not tied to a specific provider
- System must not be an onerous add-on, but be *readily usable* by normal people

Data

- *Collections*: a group of records identified by the public key of a key-pair
- *Records*: application-specific meta-information identified by the collection holding the data, the data type and a name

Collections

get (collectionID)

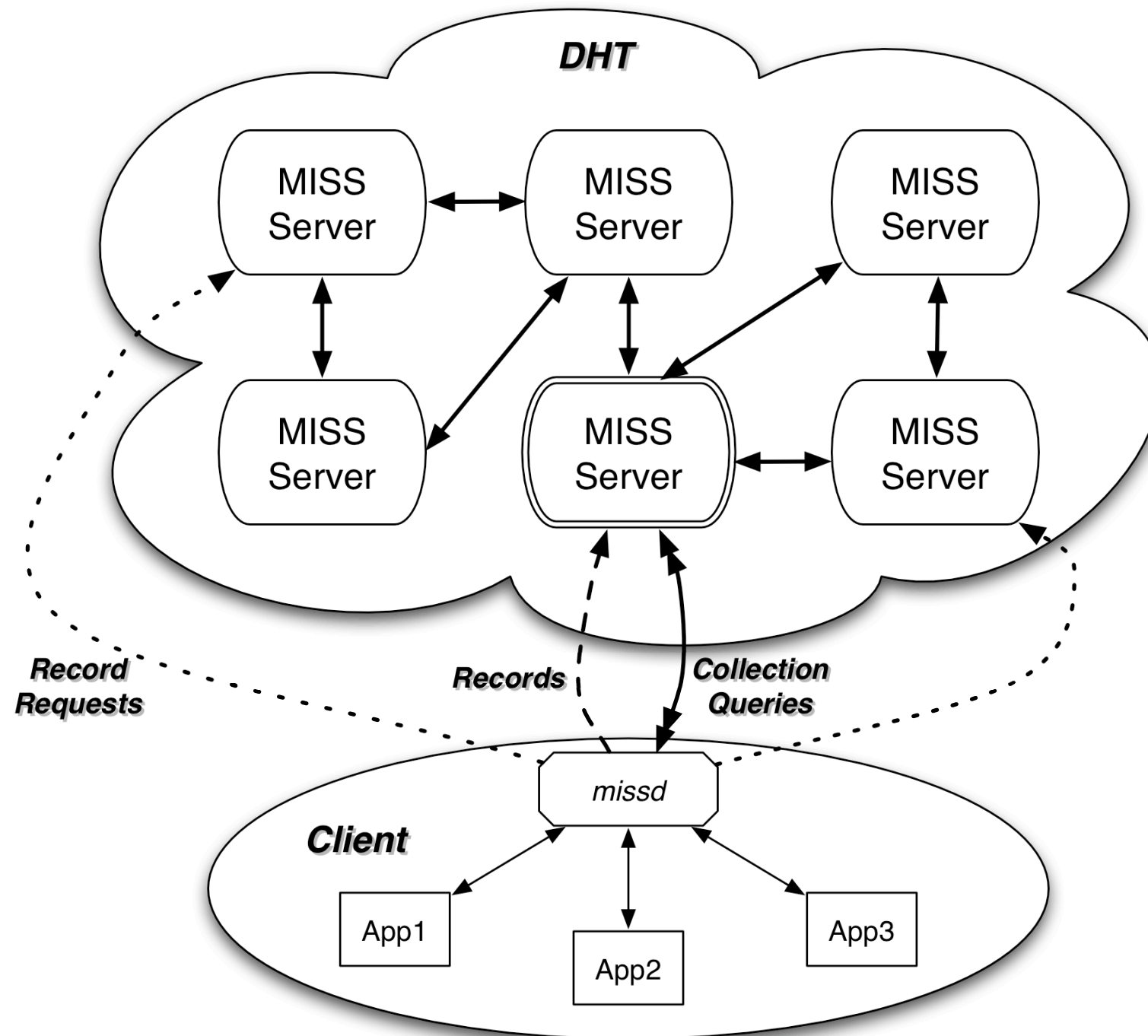
```
<miss_record>
  <type>miss</type>
  <ttl>3456</ttl>
  <signature>
    [...]
  </signature>
  <miss_collection>
    <owner>
      <name>Joe Smith</name>
      <email>jsmith@foo.com</email>
    </owner>
    <pub_key> [...] </pub_key>
    <server>128.1.2.3</server>
    <server>132.25.30.35</server>
    <server>68.45.100.7</server>
  </miss_collection>
</miss_record>
```

Records

get (CollectionID+DataType+Name)

```
<miss_record>
  <type>frob</type>
  <ttl>8311</ttl>
  <signature>
    [...]
  </signature>
  <frob>
    <ex1>foo.example.com</ex1>
    <ex2>userA</ex2>
  </frob>
</miss_record>
```

MISS Data Store



The End (Part II)

- First cut design of a system that meets the requirements in a reasonable way
- note, lots of design details worked out, but elided from talk

Applications

- OK, but what might we use this for?

Naming

- Naming is a mess
- Problem #1: names are obtuse
- Problem #2: names are hard to share
- Problem #3: names are globally unique, but ambiguous to people
- Problem #4: names are intolerant of location change
- Problem #5: naming is a distributed activity

Naming (cont.)

- Use MISS to identify other namespaces (== people)
 - Dad == 0x894af9c0b
- Build a *meta-naming layer* within MISS collections
 - Dad:email == wes.allman@some.isp.com
 - Bob:vacation-pix == http://www.flickr.com/...

Relationships

- Move social graphs outside of particular applications
 - and, outside the application layer all together
- Develop a rich set of relationship types
 - known, set, partner, delegate, observed (etc.)

Relationships (cont.)

- Allow relationship information to be used pervasively throughout the system
 - E.g., for access control
 - E.g., to delegate a task to another host while offline
 - E.g., to check the validity of some service against experience within a social network
 - E.g., for “content networking” where the focus is on the data not its location
- Achilles heel: privacy

User-Directed Protocols

- Allow users to direct the system to process their traffic in particular ways
 - E.g., pipe all email through some filter
 - E.g., define mirrors for content
 - E.g., specify redundancy for transmission
 - E.g., keeping application state
 - E.g., exposing credentials

The Actual End!

- Users are accruing meta-information at an increasingly high rate
 - rather than each app handling it on its own, let's think about a meta-information store as a basic building block
- We have designed a system that provides a reasonable meta-information store
- There are a number of examples of architectural components that could benefit from such a store

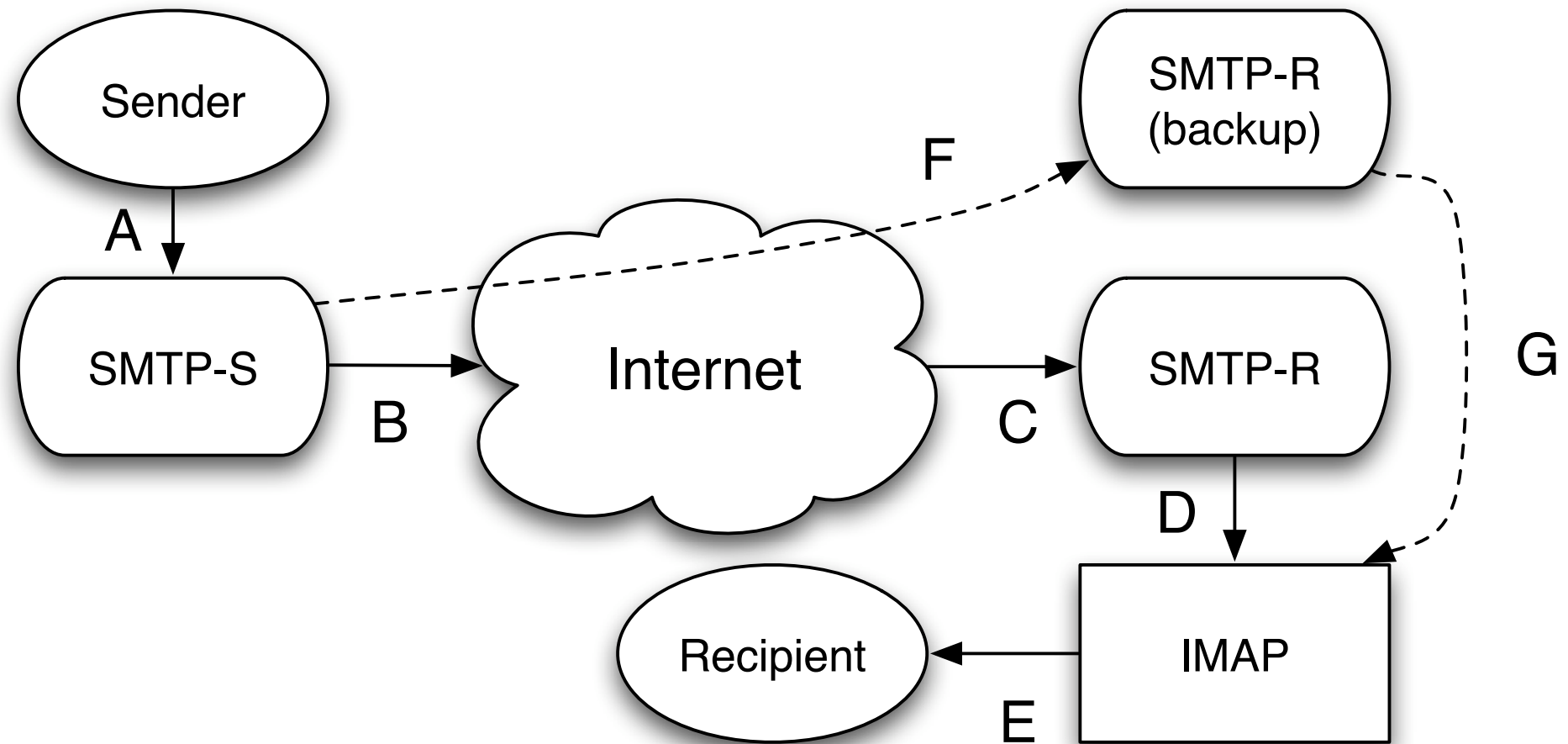


Questions? Comments?

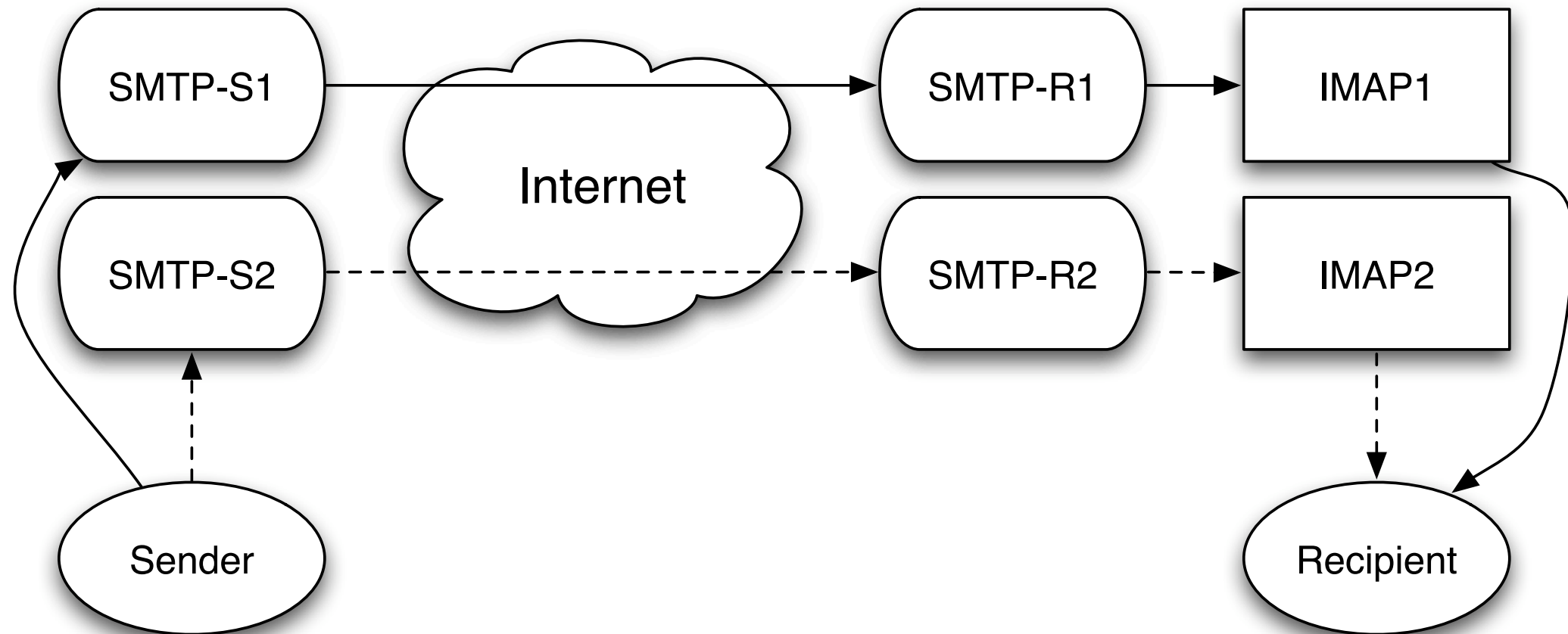
Mark Allman

mallman@icir.org
<http://www.icir.org/mallman/>

Email



Email (cont.)



MISS Data Store

