Instant Messaging

Internet Technologies and Applications

Contents

- Instant Messaging and Presence
- Comparing popular IM systems
  - Microsoft MSN
  - AOL Instant Messenger
  - Yahoo! Messenger
- Jabber, XMPP and Google Talk

Internet Messaging

- Email
  - Asynchronous communication: user does not have to be online for message to be delivered (not *instant* messaging)
- Newsgroups
- Instant Messaging and Presence
  - UNIX included finger and talk
    - Finger: determine the presence (or status) of other users
    - Talk: text based instant chatting application
  - Internet Relay Chat (IRC)
    - Introduced in 1988 as group based, instant chatting service
    - Users join a chat room
    - Networks consist of servers connected together, and clients connect via a single server
  - ICQ (“I Seek You”)
    - Introduced in 1996, allowing chatting between users without joining chat room
    - In 1998 America Online (AOL) acquired ICQ and became most popular instant messaging application/network
  - AIM, Microsoft MSN, Yahoo! Messenger, Jabber, …
    - Initially, Microsoft and Yahoo! Created clients to connect with AIM servers
    - But restricted by AOL, and most IM networks were limited to specific clients
    - Only recently (1-2 years) have some IM networks opened to different clients

Instant Messaging and Presence

- Instant Messaging
  - Synchronous communications: message is only sent to destination if recipient is willing to receive it at time it is sent
- Presence
  - Provides information about the current status/presence of a user to other users
  - Other users can subscribe to the presence information of a particular user
  - E.g.: Online, Busy, Away, Offline
  - Controls what messaging options are available (cannot send message when someone is offline)
- Naming
  - Most systems use email address format for naming

Popular IM Applications/Networks

- Closed – use proprietary protocols, normally limiting only their own clients to access network
  - AOL Instant Messaging (AIM)
  - Microsoft Messenger (MSN)
    - Also known as Windows Messenger, Live Messenger
  - Yahoo! Messenger (YMSG)
- Open – use open (published) protocols, normally allowing any client to access network
  - Jabber and XMPP

Comparing AIM, MSN and YMSG

	AIM	YMSG	MSN
Binary-based protocol	Y	Y	N
ASCII-based protocol	N	N	Y
Supports P2P connections	Y	Y	N
Rate-limiting support	Y	Y	N
User-created public chat rooms	N	Y	Y

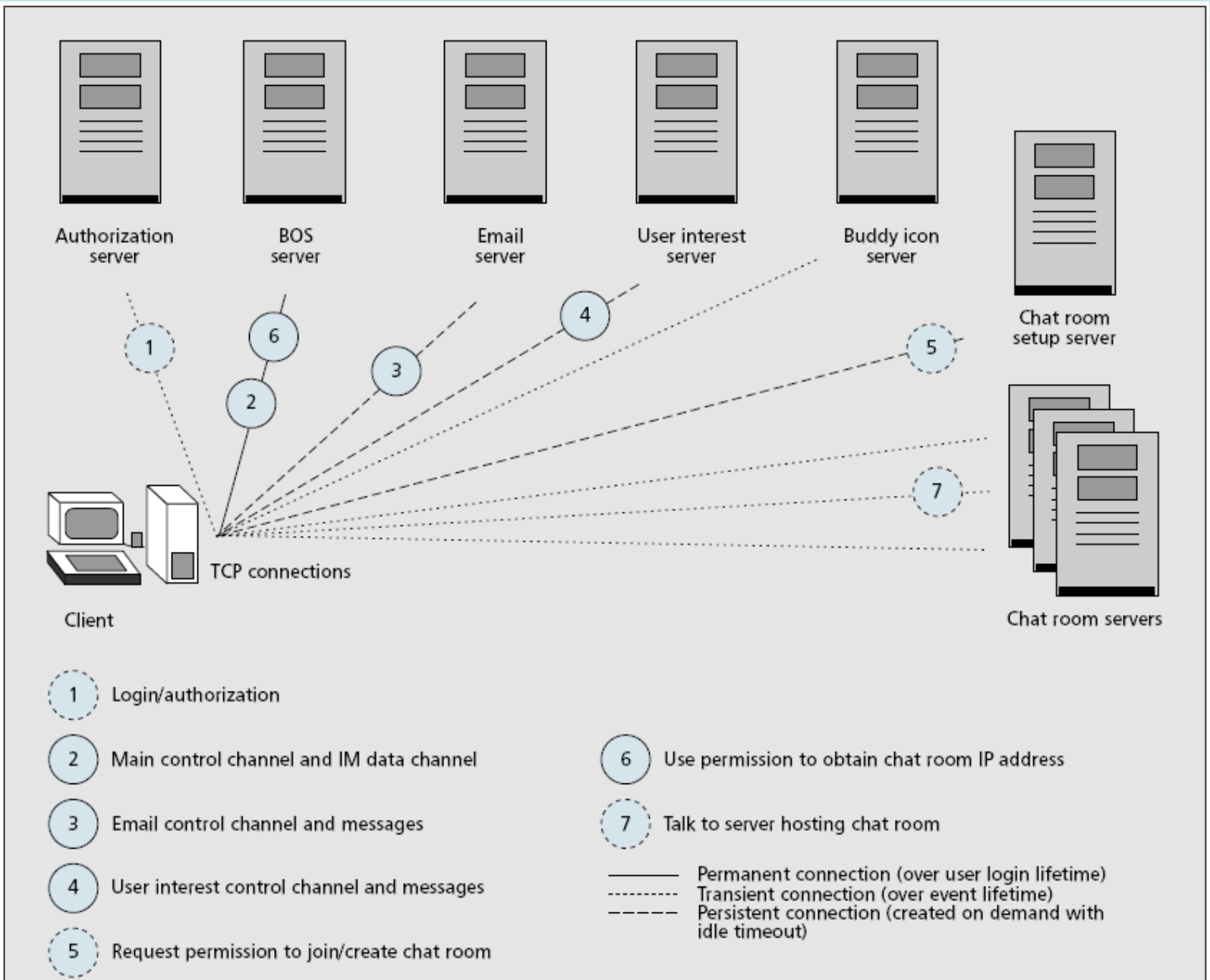
Client/Server Architecture

- A client/server architecture is main mode of operation for AIM, MSN and YMSG
  - Provider (e.g. AOL, MS, Yahoo) host servers for network
  - Clients connect to servers to establish sessions and exchange messages
  - How to make system scale with number of users?
    - Symmetric servers (YMSG)
      - Each server performs identical (and all) functions, but they are replicated
      - Users will log in to one of the servers based on random selection, geographical/network proximity, load, etc.
    - Asymmetric servers (MSN, AIM)
      - Each server is dedicated to a particular activity such as log on, basic messaging, chat room, presence, …
      - Users contact the necessary server
      - Log in server uses well known port/address to connect to
  - Advantage of Client/Server architecture: providers can easily control what users do; easier for clients to access services via firewalls
  - Disadvantage of Client/Server architecture: scaling service as number of users increases is difficult – as a result, several services are offered in P2P mode (e.g. voice-chat sessions)
    - MSN and AIM use P2P for voice chat; however it is much harder to do group-based voice communications using this model

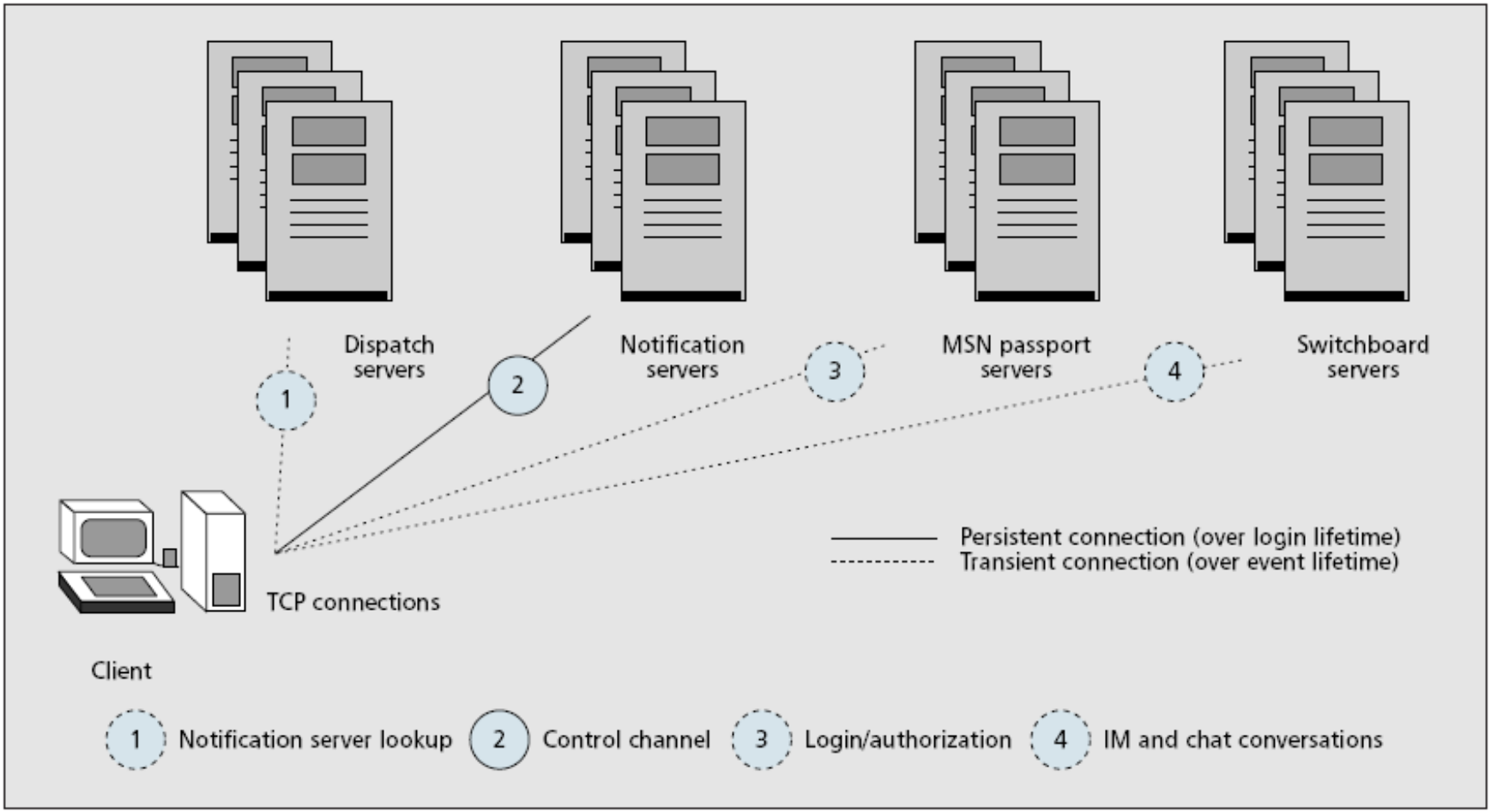
Features and Functions

- Maintaining lists of friends (and enemies)
  - Buddy list: users who are considered friends; notified of presence of these users
  - Block list: users who cannot contact you
  - Allow list: users that can contact you
  - Reverse forward lists: users that have you on their allow list
  - All lists are maintained on provider servers; synchronised when a user logs in
- Messages to describe user's current typing activity:
  - E.g. typing, not typing, typed but erased
- Deliver messages to users that are not online (similar approach to how email works)
- Secure communications (at least offered by AIM)
  - SSL used to secure messages and chat rooms

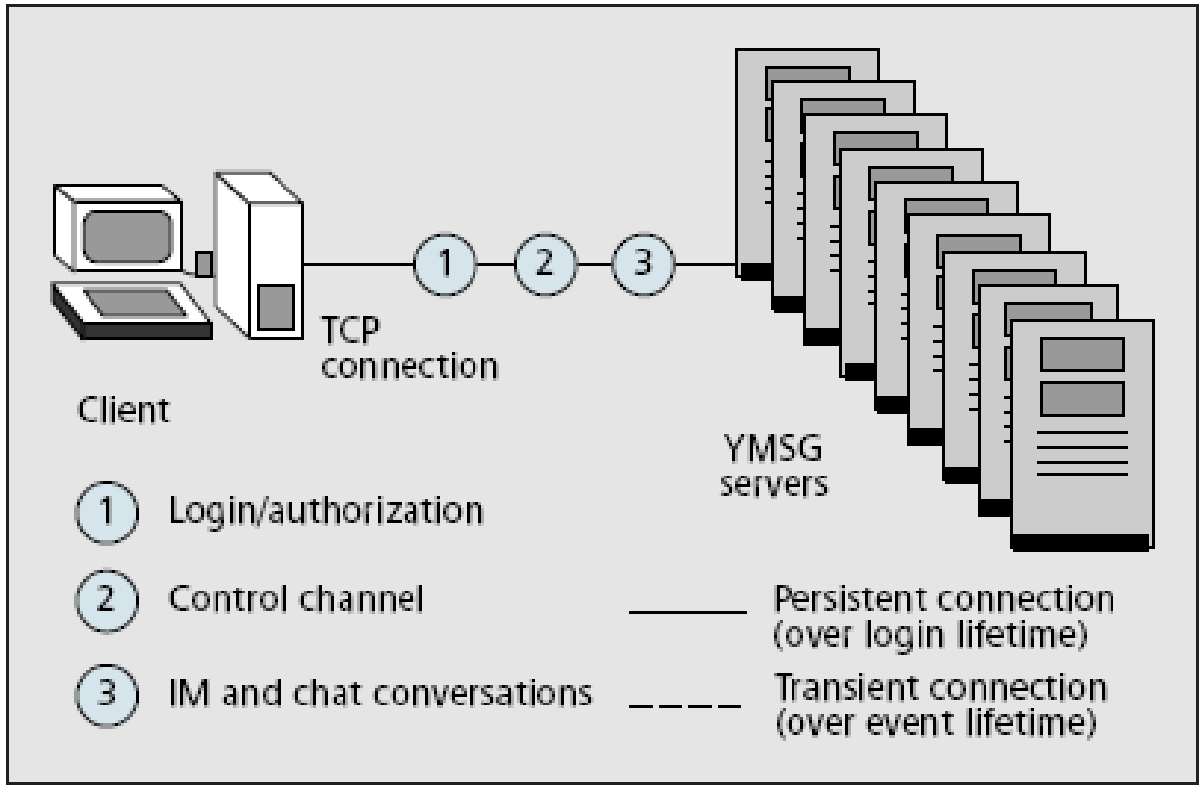
AIM System Architecture



MSN System Architecture



Yahoo! System Architecture



Session Distribution

- How do systems distribute functionality/session across multiple servers?
- AIM
  - Login to main authentication service, which directs client to BOS server
  - BOS = Basic OSCAR Service; OSCAR is a basic messaging protocol, originally develop in ICQ
  - TCP connection with BOS server is established, and this is main connection for exchanging information with provider, as well as text instant messages
    - BOS server provides addresses of other services, and client connects them on demand
    - To access chat room, client finds address of chat room setup server; chat room setup server sends verification that client can access a chat room to BOS server; then BOS server directs client to a specific chat room server
- Yahoo!
  - A single TCP connection is used to handle all control messages, instant messages and chat sessions

Session Distribution

- MSN
  1. Client initially contacts dispatch server (which has well-known address/port)
  2. Client directed to notification server and establishes a permanent TCP connection
     - Used for main control messages, including presence notifications
  3. Authentication is performed with MSN passport servers (described shortly)
  4. All IM and chat sessions are via switchboard servers
     - IM and chat are treated the same (IM is just chat between 2 people)
     - Also handles requests for file transfer, voice/video sessions (which are then established peer-to-peer between clients)

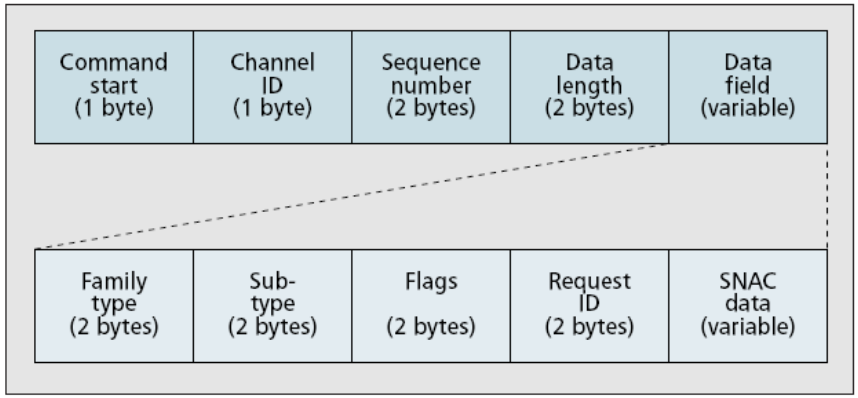
User Authentication

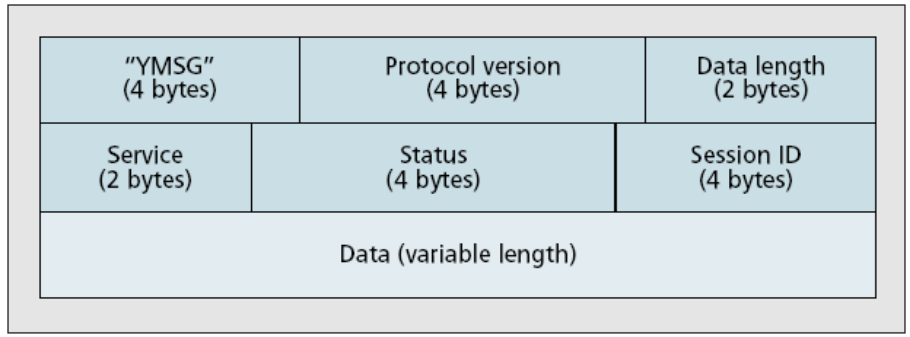
- User and provider have a shared secret: user's password
- Login authentication performed using HTTP over TLS
  - A hash of password is sent (so others cannot see password)
    - Although username is sent in clear
    - Weaknesses in some hash algorithms may allow a dictionary attack on the password
  - Avoids using expensive (in computation) public key encryption operations
- MSN and AIM send cookies (in clear) to client as credentials
  - Possible for attacker to intercept cookie and then impersonate the client

Data Transfer

- Message formats
  - AIM and YMSG use binary formats, which are more efficient than text formats
    - AIM uses variable length headers, which can be efficient
    - YMSG uses fixed length headers, which are easier to parse and process

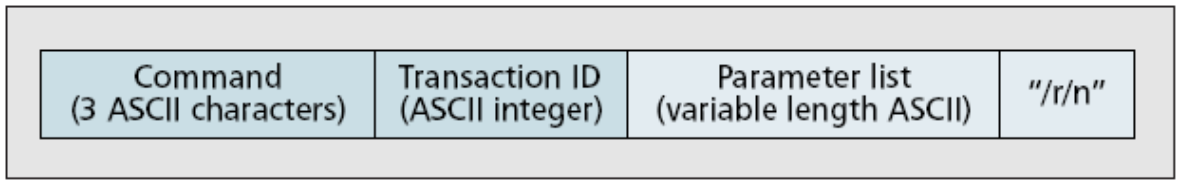
AIM





YMSG

  - MSN uses text format, which is easier to understand and debug (for humans)



MSN

Data Transfer

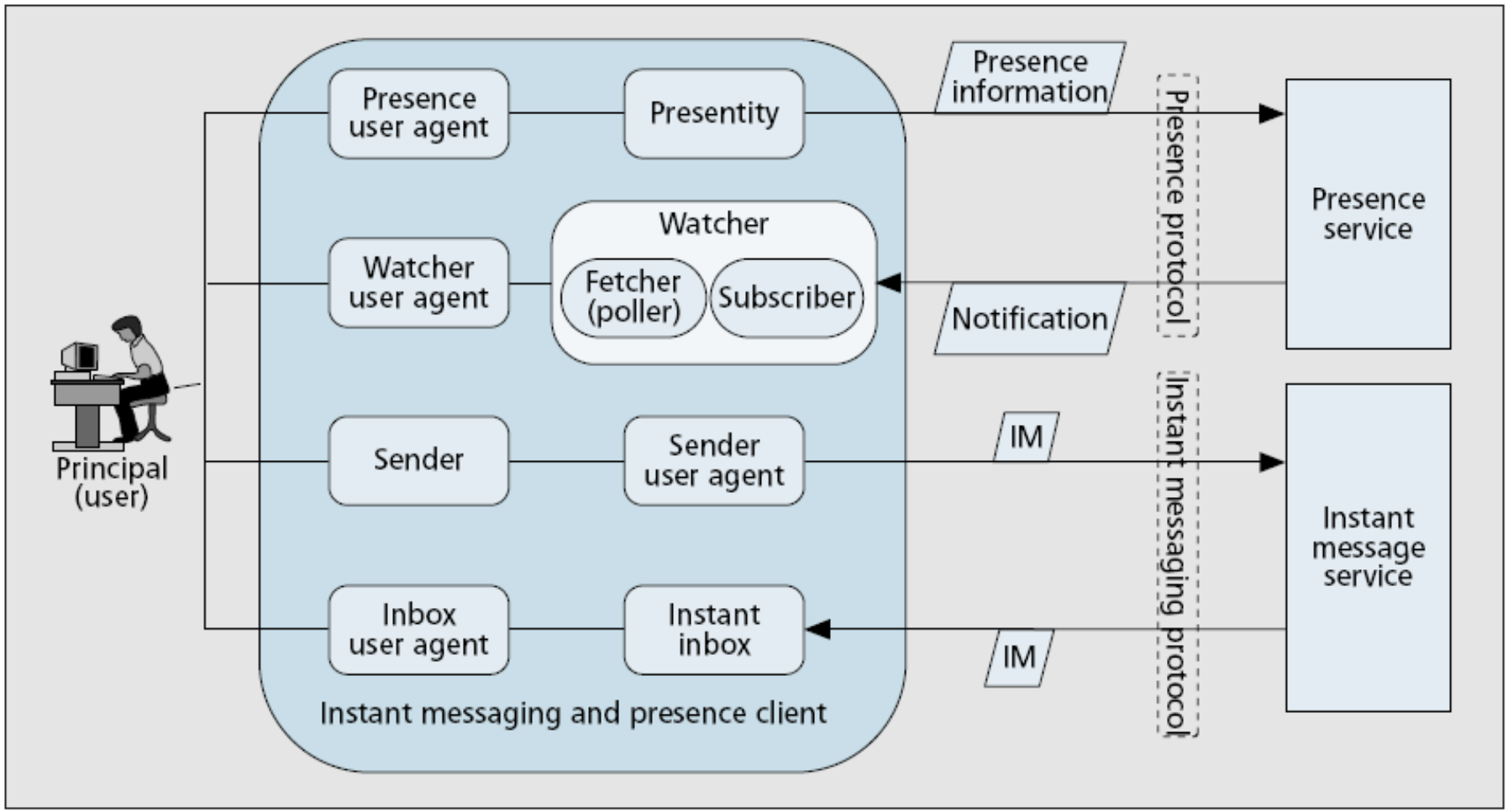
- Excessive message rates
  - With centralised servers, an IM network could easily be flooded if users send IMs at high rates
  - TCP provides congestion control in Internet, but IM providers also need to protect servers against denial-of-service attacks
  - AIM provides rate control on different message times
    - If number of messages over a period of time exceeds threshold, users' are warned, and may be disconnected
  - YMSG has a static limit of messages per second (e.g. 3)
    - Controlled by client provided by Yahoo! Therefore other clients may not implement this rate control
- Session lifetimes
  - When client is inactive, need to end session (TCP connections) as they consume memory and CPU resources
  - All systems use a heartbeat – periodic messages to keep session alive
    - AIM: client sends heartbeat every minute
    - YMSG: server sends request to client every X minutes, client must respond
    - MSN: client heartbeat and server heartbeat (client must respond)

Jabber and XMPP

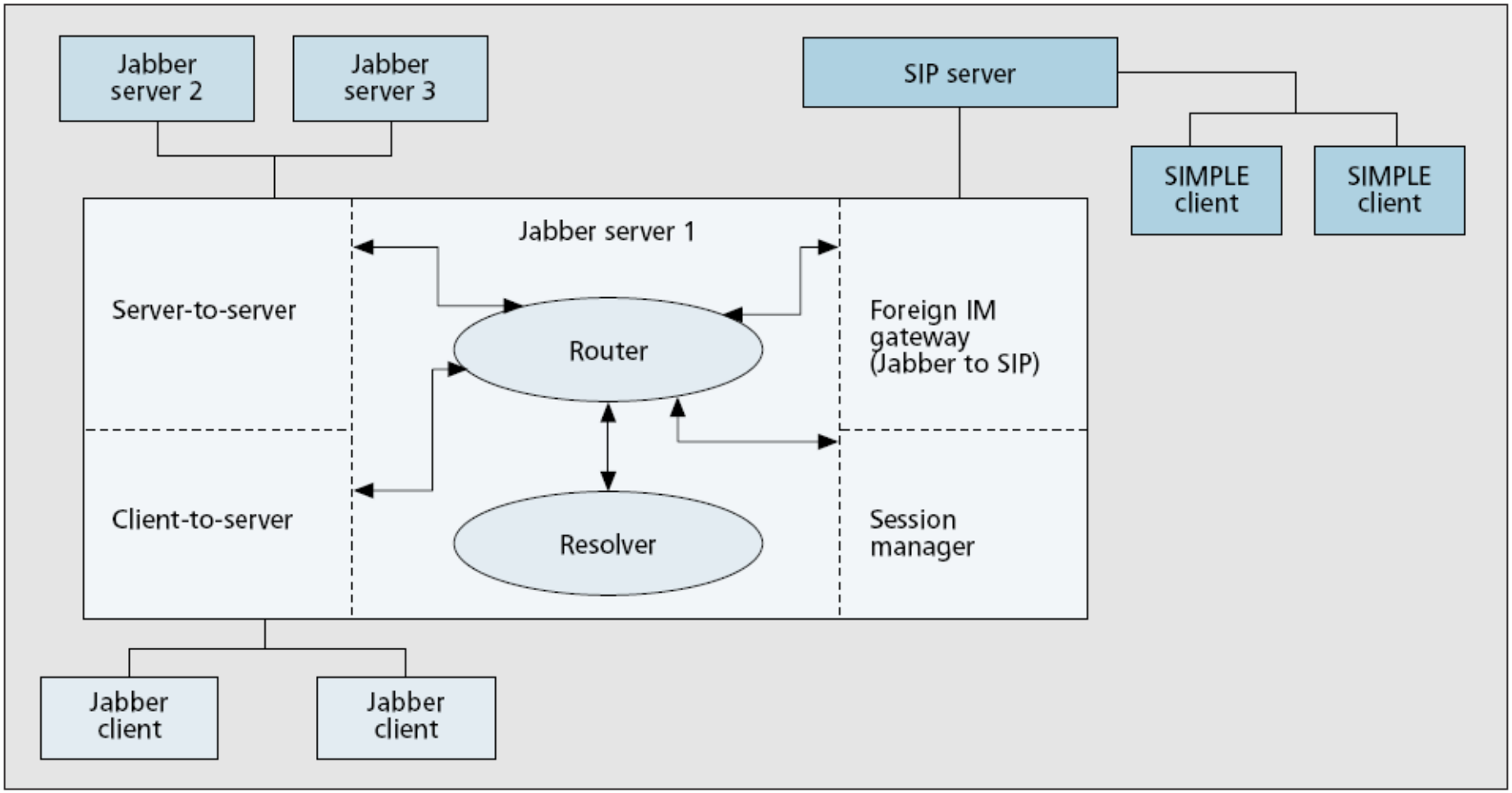
Jabber and XMPP

- Jabber was an open source IM application (and protocol) developed in 1998
- Separately IETF provided a general framework for IM in the Instant Messaging and Presence (IM&P) model (RFC2778)
  - Jabber was standardised as Extensible Messaging and Presence Protocol (XMPP) by IETF (RFC3921)
  - The IETF also had a separate Working Group that developed SIMPLE, an alternative IM protocol using Session Initiation Protocol (SIP) (RFC3428)
  - XMPP is now extended by XMPP Standards Foundation
- XMPP is used by:
  - Google Talk; Gizmo; supported by Gaim; many Jabber clients and servers

Generic IETF IM&P Model



Jabber Architecture



Jabber Architecture

- Client/Server architecture is used in Jabber
  - Although not centralised – can be many servers, and hence messages must be forwarded between servers
- Clients connect to servers using TCP
  - XMPP is used to exchange information between client and server
- Server:
  - Resolver determines where to send a message
  - Router routes/forwards the message based on info from resolver
  - Servers exchange information with other servers using XMPP
  - Stores clients information and contact lists
- Gateway may be implemented to connect to other IM systems, e.g. SIP/SIMPLE, MSN, AIM, …

XMPP

- Protocol based on XML messages
  - Three defined XML message types:
    - Message – carries IMs between clients and servers
      - chat, error, groupchat, headline, normal
    - Presence – used to notify client about status of users
      - unavailable, subscribe, subscribed, unsubscribe, unsubscribed, probe, error
    - IQ (info/query) – request/response queries to exchange other information between clients and servers (e.g. setting up a session)

Frame — Time of packet arrival, total size in bytes (311 bytes).			
Ethernet (14 bytes) — MAC addresses of the destination and source	Internet Protocol (20 bytes) — Version of IP, type of protocol	Transmission Control Protocol (20 bytes) — Source port, destination port, window size, checksum	Jabber XML Messaging (257 bytes) — <message type='chat' to= 'bob@foobar.com'><x xmlns='jabber:x:event'> <composing/></x><body> Hello!</body><html xmlns='http://jabber.org/protocol/xhtml-im'><body xmlns='http://www.w3.org/ 1999/xhtml'>Hello!</body> </html></message> (If this message is prefixed with "emoticon" of smile it will be represented as - ":-) Hello" and the total number of bytes will increase by 3.)

XMPP/Jabber Example Messages

- Client to server:

```
<?xml version="1.0"?>
<stream:stream xmlns:stream=http://etherx.jabber.org/streams
 xmlns="jabber:client" to="example.org">
```

- Server response:

```
<stream:stream xmlns='jabber:client‘
xmlns:stream='http://etherx.jabber.org/streams' from='example.org'
id='1461777714'>
```

- Client login (unsecure; there is a secure option):

```
<iq type="set" id="auth_2" to="example.org" >
  <query xmlns="jabber:iq:auth">
    <username>alice</username>
    <password>password</password>
    <resource>Work</resource>
  </query>
</iq>
```

- Server result:

```
<iq from="example.org" id='auth_2' type='result'/>
```

XMPP/Jabber Example Messages

- Client message:

```
<message to="bob@example.com" >
  <subject>Hello!</subject>
  <body>Can't wait to see you tomorrow.</body>
</message>
<presence type="unavailable" >
  <status>Logged out</status>
</presence>
</stream:stream>
```

- The message is then sent to Bob (either his server or client)
- Server response to client:

```
</stream:stream>
```