Internet application protocols

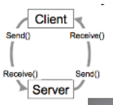
Introduction

CSCI E 45b: The Cyber World – part B

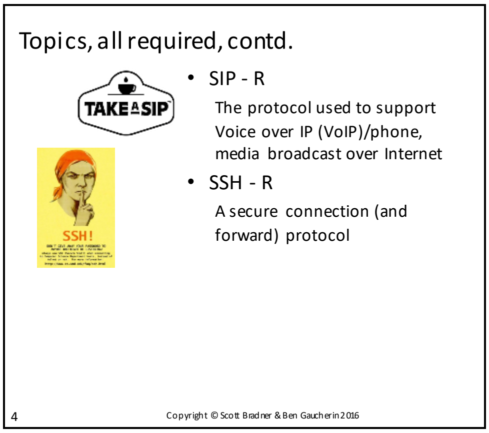
Introduction: learning goals

- Understand the basic structural elements of Internet application protocols
- Understand the basic mechanics of the most popular Internet application protocols

Topics, all required



- Base concepts - R
  Key terms and ideas common across most protocols
- Finger - R
  One of the simplest Internet application protocol
- SMTP - R
  The primary protocol used for sending and distributing email
- HTTP – R
  The protocol used for the Web



Topics, all required, contd.

- SIP - R

  The protocol used to support Voice over IP (VoIP)/phone, media broadcast over Internet
- SSH - R

  A secure connection (and forward) protocol

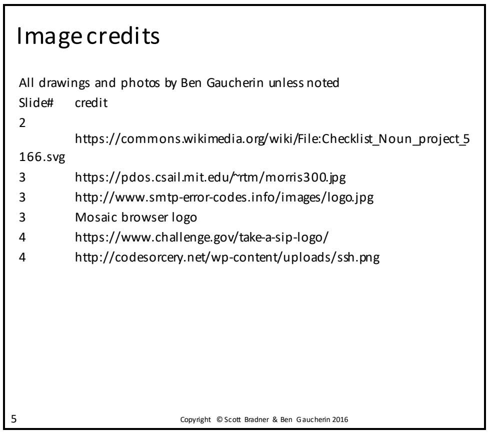


Image credits

All drawings and photos by Ben Gaucherin unless noted

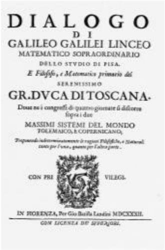
Slide#	credit
2	https://commons.wikimedia.org/wiki/File:Checklist_Noun_project_5166.svg
3	https://pdos.csail.mit.edu/~rtm/morris300.jpg
3	http://www.smtp-error-codes.info/images/logo.jpg
3	Mosaic browser logo
4	https://www.challenge.gov/take-a-sip-logo/
4	http://codesorcery.net/wp-content/uploads/ssh.png

Internet application protocols

Base concepts

CSCI E 45b: The Cyber World – part B

Application protocols



DIALOGO
DI
GALILEO GALILEI LINCEO
MATEMATICO SOPRAORDINARIO
DELLO STVDIO DI PISA.
E Filoſofo, e Matematico primario del
SERENISSIMO
GR.DVCA DI TOSCANA.
Doue ne i congreſſi di quattro giornate ſi diſcorre
ſopra i due
MASSIMI SISTEMI DEL MONDO
TOLEMAICO, E COPERNICANO;
Proponendo indeterminatamente le ragioni Filoſofiche, e Naturali
tanto per l'una, quanto per l'altra parte.
CON PRI VILEGI.
IN FIORENZA, Per Gio:Batiſta Landini MDCXXXII.
CON LICENZA DE' SVPERIORI.

- The structures, and sequence of network messages required to implement an application
- And the actions that create these messages or result from the receipt of these messages

e.g. email, viewing documents, connecting to a host, etc.

Basic mechanics of an application

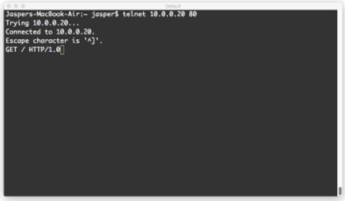
Finger	79
HTTP	80
SSH	22
SMTP	25
SIP	5060

- Listens on a predetermined port – default application port
- Expects a predetermined protocol to be used – the application protocol
- Most IETF (and thus most Internet) protocols are plain text protocols

Basic mechanics of an application, contd.

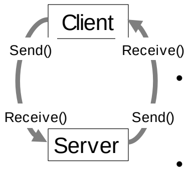
- You can use the Telnet command to test many network services/applications:

```
telnet [host [port]]
```



```
Jaspers-MacBook-Air:~ jasper$ telnet 10.0.0.20 80
Trying 10.0.0.20...
Connected to 10.0.0.20.
Escape character is '^]'.
GET / HTTP/1.0
```

Client, Server, or both?

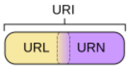


- Client – the machine/software from which requests originate
- Server – the machine/software fulfilling the request
- A machine often is both client and server – the role is context dependent
- Peer to Peer

  No central server required

Names and identifiers

www.xyz.com
www.xyz.org
www.xyz.net
www.xyz.gov
www.xyz.mil
…



- IP address is the identifier and locator of a node on the network
- Resolving names - DNS

  A records – map a user friendly name to an IP address

  CNAME records – link a name to another name

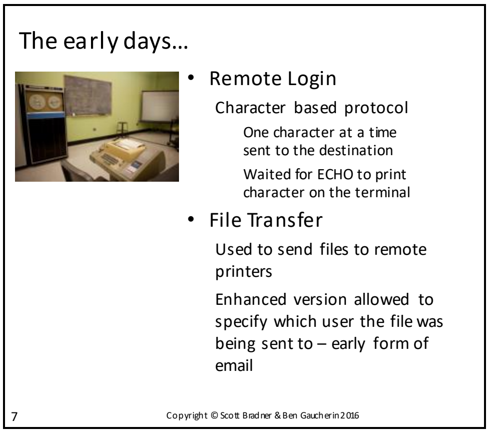
  MX records – identify mail servers for a domain
- URI: URN/URL

  Uniform Resource Locator (URL) - includes a domain name

  e.g., http://www.harvard.edu

  Uniform Resource Name (URN)

  e.g. urn:isbn:0-300-15124-1



The early days…

- Remote Login
  - Character based protocol
    - One character at a time sent to the destination
    - Waited for ECHO to print character on the terminal
- File Transfer
  - Used to send files to remote printers
  - Enhanced version allowed to specify which user the file was being sent to – early form of email

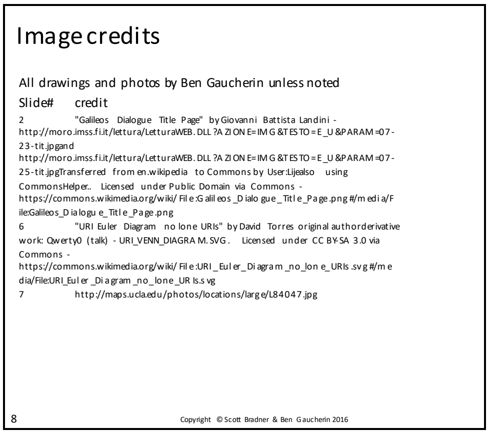


Image credits

All drawings and photos by Ben Gaucherin unless noted

Slide#	credit
2	"Galileos Dialogue Title Page" by Giovanni Battista Landini - http://moro.imss.fi.it/lettura/LetturaWEB.DLL?AZIONE=IMG&TESTO=E_U&PARAM=07-23-tit.jpgand http://moro.imss.fi.it/lettura/LetturaWEB.DLL?AZIONE=IMG&TESTO=E_U&PARAM=07-25-tit.jpgTransferred from en.wikipedia to Commons by User:Lijealso using CommonsHelper.. Licensed under Public Domain via Commons - https://commons.wikimedia.org/wiki/File:Galileos_Dialogue_Title_Page.png#/media/File:Galileos_Dialogue_Title_Page.png
6	"URI Euler Diagram no lone URIs" by David Torres original authorderivative work: Qwerty0 (talk) - URI_VENN_DIAGRAM.SVG. Licensed under CC BY-SA 3.0 via Commons - https://commons.wikimedia.org/wiki/File:URI_Euler_Diagram_no_lone_URIs.svg#/media/File:URI_Euler_Diagram_no_lone_URIs.svg
7	http://maps.ucla.edu/photos/locations/large/L84047.jpg

Internet application protocols

Finger

CSCI E 45b: The Cyber World – part B

History

Robert Morris Jr.

- Early days protocol to get information about users from a machine on the Internet
- Very simple protocol specified in 1977 in RFC-742
- Used by the Morris Worm
- Security risk, not in use a whole lot – Duh!

For fun you can try

```
finger @bathroom.mit.edu*
```

* Not always responsive

Before the worm, Morris setup the Finger service on HARV-10 as a "mirror Finger"

How it works

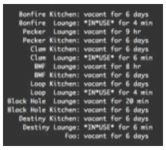
- The command is issued on the client machine

```
finger [user] [user@host]
[@host]
```

- The finger utility sends a packet to the target (server) machine's port 79

How it works, contd.

- The payload of the packet contains either:
  - An empty line – get information on all users on the target machine
  - A user name – get information about a specific user
- The target machine responds with text detail on the user(s)



```
   Bonfire Kitchen: vacant for 6 days
   Bonfire  Lounge: *IN*USE* for 4 min
    Pecker  Lounge: vacant for 9 hr
    Pecker Kitchen: vacant for 6 days
      Clam Kitchen: vacant for 6 days
      Clam  Lounge: *IN*USE* for 6 min
       BMF  Lounge: vacant for 8 hr
       BMF Kitchen: vacant for 6 days
      Loop Kitchen: vacant for 6 days
      Loop  Lounge: *IN*USE* for 4 min
Black Hole  Lounge: vacant for 20 min
Black Hole Kitchen: vacant for 6 days
   Destiny Kitchen: vacant for 6 days
    Destiny Lounge: *IN*USE* for 6 min
               Foo: vacant for 6 days
```

Image credits

All drawings and photos by Ben Gaucherin unless noted

Slide#	credit
2	https://pdos.csail.mit.edu/~rtm/morris300.jpg

Internet application protocols

Small Mail Transfer Protocol (SMTP)

CSCI E 45b: The Cyber World – part B

History

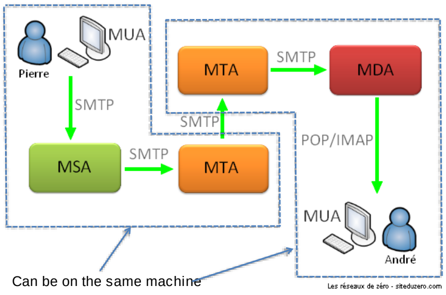
- SMTP formally defined in 1982 by RFC-821, although email was in use since the early 70's
- In the early days of email
  - Mail messages were sent as individual files to a specific user

- The first email was sent from BBN-TENEXB (back) and received by BBN-TENEXA (front) – both machines being side-by-side

Mail processing model

- SMTP is for sending files in mail format, not retrieving email
  - Mail User Agent (MUA) – mail client software
  - Mail Submission Agent (MSA) – sends messages received by MUAs
  - Mail Transfer Agent (MTA) – relays (send/receive) email using SMTP
  - Mail Delivery Agent (MDA) – collects messages destined for local users and makes them available to their MUA

Mail processing model, contd.



Can be on the same machine

Les réseaux de zéro - siteduzero.com

A simple SMTP exchange

```
<connection made on port 25>
220 frontend01.harvard.edu ESMTP Server
ready
HELO test.com
250 OK
MAIL FROM: testuser@test.com
250 OK - mail from <testuser@test.com>
RCPT TO: administrator@frontend01.harvard.edu
250 OK - Recipient
<administrator@frontend01.harvard.edu>
DATA
354 Send data. End with CRLF.CRLF
Hi there!  How are you?


250 OK
QUIT
221 closing connection
```

Status codes

- **220** Service ready
- **221** Service closing transmission channel
- **250** Requested mail action okay, completed
- **251** User not local; will forward to
- **354** Start mail input; end with "."
- **4xx** and **5xx** series Error situations of various kinds
  E.g. **550** – No such user

MIME and SMTP/MIME

- Original SMTP supported sending 7 bit ASCII coded characters
- MIME - Multipurpose Internet Mail Extensions
  - Defines a mechanism for sending other types of data over email
- Now MIME is used by other protocols for the same purpose

MIME and SMTP/MIME, contd.

- Primary headers (with sample values):

```
MIME-version: 1.0
Content-Type: text/plain
Content-Transfer-Encoding:
<7bit|8bit|base64|binary|qu
oted-printable>
```

MIME and SMTP/MIME, contd.

- Multipart messages

```
Content-Type:
multipart/mixed;
boundary=<boundary>
```

  - To send different content-types in the same message
  - To send large messages (e.g. files)
  - This allows the sending of messages of any size
    - But MUA and MSAs, MDAs sometime set limits

A sample message header

```
Received: from ENTWEDGE0000001.university.harvard.edu (10.35.2.152) by
 ENTWHUBT0000004.university.harvard.edu (10.32.208.50) with Microsoft SMTP
 Server (TLS) id 14.3.146.0; Sun, 19 Jan 2015 10:28:47 -0500
Received: from ackroyd.harvard.edu (128.103.208.29) by
 ENTWEDGE0000001.university.harvard.edu (10.35.2.152) with Microsoft SMTP
 Server id 14.3.146.0; Sun, 19 Jan 2015 10:28:30 -0500
Received: by ackroyd.harvard.edu (Postfix)     id ACBC5E9A67; Sun, 19 Jan
2015
 10:28:46 -0500 (EST)
Received: from sobco.sobco.com (unknown [136.248.127.164]) by
 ackroyd.harvard.edu (Postfix) with ESMTP id 953ABE9A65 for
 <ben_gaucherin@harvard.edu>; Sun, 19 Jan 2015 10:28:46 -0500 (EST)
Received: from localhost (localhost [127.0.0.1]) by sobco.sobco.com
(Postfix)
 with ESMTP id 069915A627F for <ben_gaucherin@harvard.edu>; Sun, 19 Jan
2015
 10:28:46 -0500 (EST)
X-Virus-Scanned: amavisd-new at sobco.com
Received: from sobco.sobco.com ([127.0.0.1]) by localhost (sobco.sobco.com
 [127.0.0.1]) (amavisd-new, port 10024) with ESMTP id rQqF3W0xtX4V for
 <ben_gaucherin@harvard.edu>; Sun, 19 Jan 2015 10:28:44 -0500 (EST)
Received: from golem.sobco.com (golem.sobco.com [136.248.127.162]) by
 sobco.sobco.com (Postfix) with ESMTPSA id DDD4A5A6271 for
 <ben_gaucherin@harvard.edu>; Sun, 19 Jan 2015 10:28:44 -0500 (EST)
```

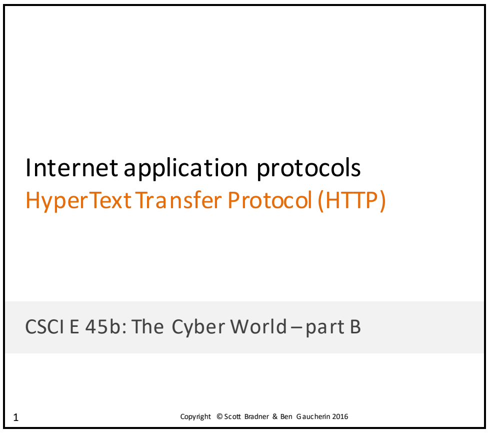
A sample message header, contd.

```
From: "Scott O. Bradner" <sob@sobco.com>
Content-Type: text/plain; charset="us-ascii"
Content-Transfer-Encoding: 7bit
Subject: to look at
Message-ID: <33114E8C-89A1-4F3B-BE0A-A6F9FC0409E0@sobco.com>
Date: Sun, 19 Jan 2015 10:28:44 -0500
To: Benoit Gaucherin <ben_gaucherin@harvard.edu>
MIME-Version: 1.0 (Mac OS X Mail 7.1 \(1827\))
X-Mailer: Apple Mail (2.1827)
Return-Path: sob@sobco.com
X-MS-Exchange-Organization-Antispam-Report: v=2.1 cv=TKZ8FTVa c=1 sm=1
tr=0
 a=jeTkmrT/jwcKSjCs2WwgaA==:117 a=SZwlRxzbqrU7NHadf9mkIA==:17
 a=bThMrJPUCi0A:10 a=wPDyFdB5xvgA:10 a=kj9zAlcOel0A:10 a=zA0KW45MAAAA:8
 a=sVy1UYKO7ONOnZoQoFgA:9 a=CjuIK1q_8ugA:10 a=diyHgU4cqYEA:10
 a=jVLyn9cCbSIA:10 a=3B-TjL6AeNYA:10 a=HXrLMtXIuHoA:10
 a=l1LbC1KgUbIA:10;OrigIP:128.103.208.29;SCL:0
X-MS-Exchange-Organization-AVStamp-Mailbox: MSFTFF;1;0;0 0 0
X-OrganizationHeadersPreserved: ENTWEDGE0000001.university.harvard.edu
X-MS-Exchange-Organization-SCL: 0
X-MS-Exchange-Organization-AuthSource:
ENTWEDGE0000001.university.harvard.edu
X-MS-Exchange-Organization-AuthAs: Anonymous
```

Image credits

All drawings and photos by Ben Gaucherin unless noted

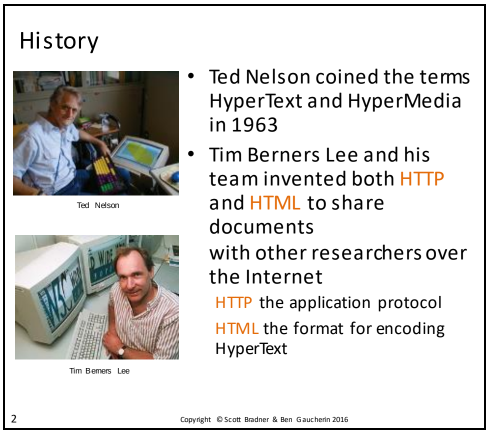
Slide#	credit
2	http://openmap.bbn.com/~tomlinso/ray/x118-17-ka10.jpg
3	http://media.nola.com/business_impact/photo/10003629-large.jpg
4	http://sdz.tdct.org/sdz/medias/uploads.siteduzero.com_files_294001_295000_294432.png
6	http://www.smtp-error-codes.info/images/logo.jpg
7-9	http://le-mime.co.uk/wp-content/uploads/2011/05/mime_2.jp g



Internet application protocols

HyperText Transfer Protocol (HTTP)

CSCI E 45b: The Cyber World – part B

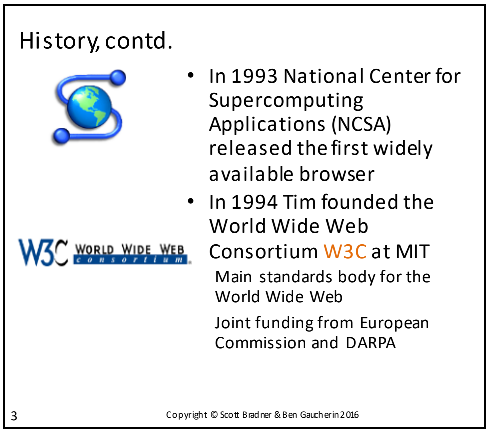


History

Ted Nelson

Tim Berners Lee

- Ted Nelson coined the terms HyperText and HyperMedia in 1963
- Tim Berners Lee and his team invented both HTTP and HTML to share documents with other researchers over the Internet
  - HTTP the application protocol
  - HTML the format for encoding HyperText

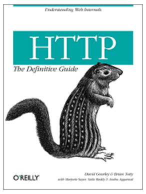


History, contd.

- In 1993 National Center for Supercomputing Applications (NCSA) released the first widely available browser
- In 1994 Tim founded the World Wide Web Consortium W3C at MIT
  - Main standards body for the World Wide Web
  - Joint funding from European Commission and DARPA

W3C WORLD WIDE WEB consortium

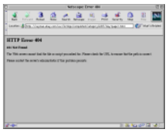
Protocol overview



- Similar to SMTP in many ways: plain text, header based, status codes
- Methods:

  GET, HEAD, POST, PUT, DELETE, TRACE, OPTIONS, CONNECT, PATCH
- Header fields – e.g. content-type

Protocol overview, contd.



HTTP Error 404

404 Not Found

- Status codes – e.g. 404 resource not found

  Note the general consistency with SMTP status codes

  2xx – Success, 3xx – Redirection, 4xx – Request error, 5xx – Application internal error

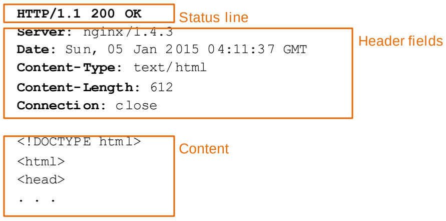
Simple HTTP exchange

<establish connection to a web server's port 80>

- Request:

```
GET /index.html HTTP/1.0
```

- Response:



```
HTTP/1.1 200 OK
Server: nginx/1.4.3
Date: Sun, 05 Jan 2015 04:11:37 GMT
Content-Type: text/html
Content-Length: 612
Connection: close

<!DOCTYPE html>
<html>
<head>
. . .
```

Getting a whole web page

- A web page is made of multiple parts:
  - Main HTML, pictures, videos, glyphs, advertisements, trackers, etc.
  - From the same server, or any number of third party servers
- Each element is obtained by a separate HTTP request
- So, a web page will typically require many (sometimes dozens) of HTTP requests to be rendered in your browser

HTTP URLs

```
<protocol>://<host><:port>/<path>?<variables>
```

<protocol> - http (or https for secure HTTP)

<host> - name or IP address of the host the request is being sent to

<:port> - defaults to 80 for HTTP, 443 for HTTPS

<path> - path to the resource (page, service, etc.)

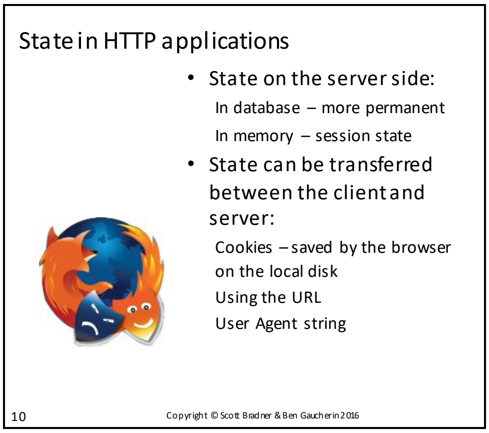
<variables> - key, value pairs to provide some state to the HTTP server

```
https://bentest.law.harvard.edu:8080/my_web_app/ind
ex.php?key1=blue&key2=vanilla
```

Mucking around with URLs

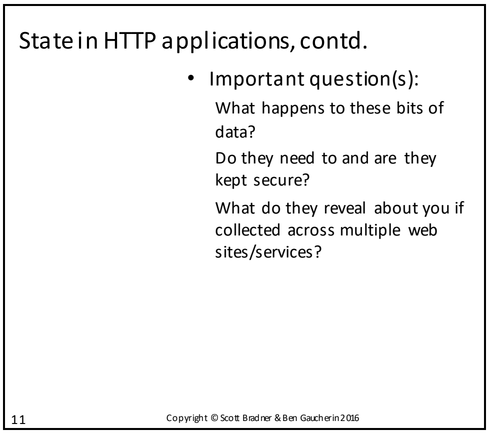
- URLs variables capture information to tailor the content you receive
  - E.g., page number in a multi-page list of results, number of lines per page
- Playing with these parameters may allow you to customize what you see
  - Sometimes in ways not intended by the website operator

GOOGLEDORKS
Google dorking is the way to query google in a way that would retrieve what you really want from Google.



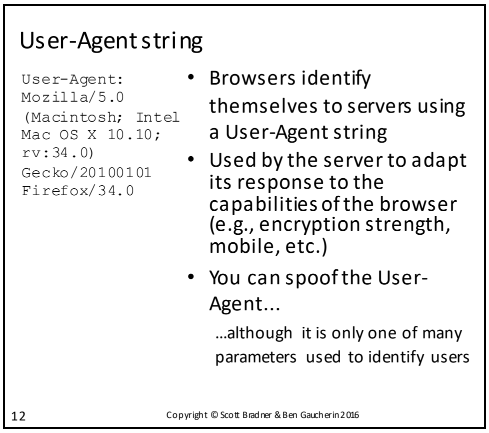
State in HTTP applications

- State on the server side:
  - In database – more permanent
  - In memory – session state
- State can be transferred between the client and server:
  - Cookies – saved by the browser on the local disk
  - Using the URL
  - User Agent string



State in HTTP applications, contd.

- Important question(s):
  - What happens to these bits of data?
  - Do they need to and are they kept secure?
  - What do they reveal about you if collected across multiple web sites/services?

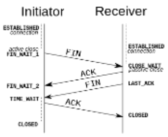


User-Agent string

```
User-Agent:
Mozilla/5.0
(Macintosh; Intel
Mac OS X 10.10;
rv:34.0)
Gecko/20100101
Firefox/34.0
```

- Browsers identify themselves to servers using a User-Agent string
- Used by the server to adapt its response to the capabilities of the browser (e.g., encryption strength, mobile, etc.)
- You can spoof the User-Agent...
  - ...although it is only one of many parameters used to identify users

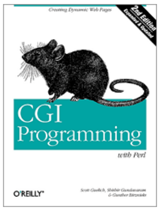
Persistent connections



- In early versions of HTTP the TCP connection was closed after each request/response exchange
- In HTTP 1.1 persistent connections (aka Keep Alive) were introduced to keep the connection open

Remember – every web page can have hundreds of individual components requiring a separate HTTP request for each

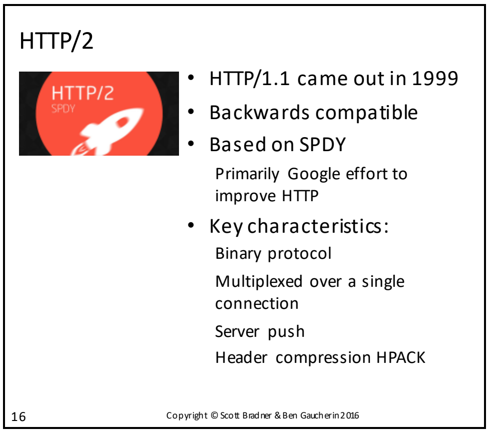
Common Gateway Interface (CGI)



- CGI and derivative technologies are what people use to generate dynamic content on the Web
- CGI is a standard for calling command line executables to generate the response to an HTTP request
- A team from the NCSA wrote the first specs
- Which evolved to become RFC 3875

HTTP Secure - HTTPS

- Initially developed by Netscape for commerce
- Not a protocol itself, but layering HTTP on top of SSL/TLS to encrypt the end-to-end HTTP exchange
- Other approaches that failed: S-HTTP was looking to encrypt just the content, not the end-to-end connection



HTTP/2

- HTTP/1.1 came out in 1999
- Backwards compatible
- Based on SPDY
  - Primarily Google effort to improve HTTP
- Key characteristics:
  - Binary protocol
  - Multiplexed over a single connection
  - Server push
  - Header compression HPACK

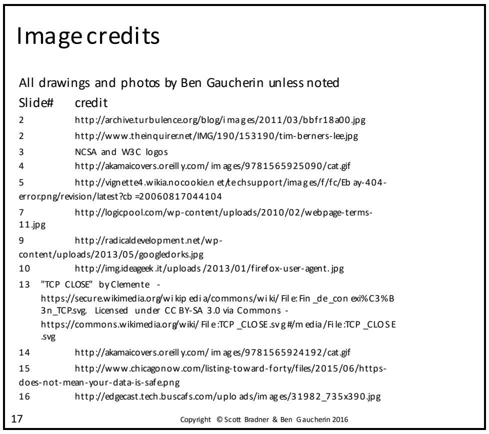


Image credits

All drawings and photos by Ben Gaucherin unless noted

Slide#	credit
2	http://archive.turbulence.org/blog/images/2011/03/bbfr18a00.jpg
2	http://www.theinquirer.net/IMG/190/153190/tim-berners-lee.jpg
3	NCSA and W3C logos
4	http://akamaicovers.oreilly.com/images/9781565925090/cat.gif
5	http://vignette4.wikia.nocookie.net/techsupport/images/f/fc/Ebay-404-error.png/revision/latest?cb=20060817044104
7	http://logicpool.com/wp-content/uploads/2010/02/webpage-terms-11.jpg
9	http://radicaldevelopment.net/wp-content/uploads/2013/05/googledorks.jpg
10	http://img.ideageek.it/uploads/2013/01/firefox-user-agent.jpg
13	"TCP CLOSE" by Clemente - https://secure.wikimedia.org/wikipedia/commons/wiki/File:Fin_de_conexi%C3%B3n_TCP.svg. Licensed under CC BY-SA 3.0 via Commons - https://commons.wikimedia.org/wiki/File:TCP_CLOSE.svg#/media/File:TCP_CLOSE.svg
14	http://akamaicovers.oreilly.com/images/9781565924192/cat.gif
15	http://www.chicagonow.com/listing-toward-forty/files/2015/06/https-does-not-mean-your-data-is-safe.png
16	http://edgecast.tech.buscafs.com/uploads/images/31982_735x390.jpg

Internet application protocols

Cookies

CSCI E 45b: The Cyber World – part B

Cookies – what it is

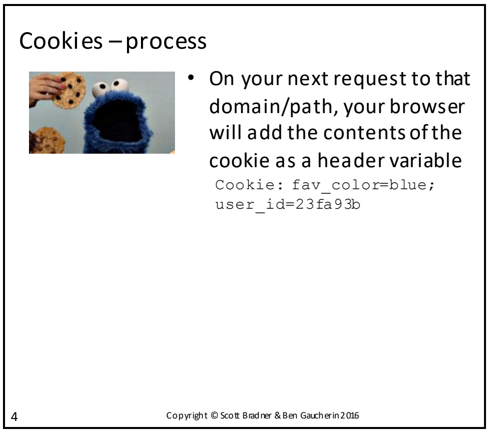
- Originally used to provide continuity of identity
  - Moving to device fingerprints rather than using cookies
- Use expanded to a wide array of scenarios:
  - Personalization, access counts
- Cookie access is limited to the domain that set the cookie
- Some cookies are encrypted

Cookies – process

- When you visit, the server sends a set-cookie command as part of its response header

```
Set-Cookie:
user_id=23fa93b;
Expires=Sat, 25 Jun 2015
11:00:00 GMT
```

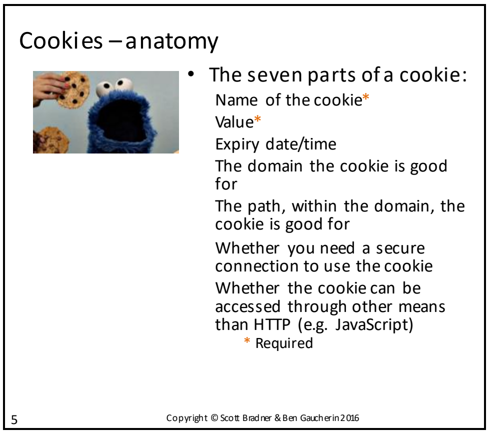
- The cookie is saved on your machine, if your browser is set to accept cookies



Cookies – process

- On your next request to that domain/path, your browser will add the contents of the cookie as a header variable

```
Cookie: fav_color=blue;
user_id=23fa93b
```



Cookies – anatomy

- The seven parts of a cookie:

  Name of the cookie*

  Value*

  Expiry date/time

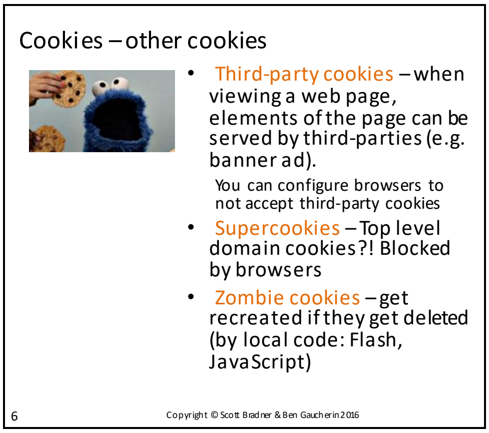
  The domain the cookie is good for

  The path, within the domain, the cookie is good for

  Whether you need a secure connection to use the cookie

  Whether the cookie can be accessed through other means than HTTP (e.g. JavaScript)

  * Required

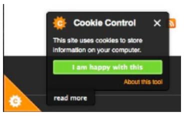


Cookies – other cookies

- Third-party cookies – when viewing a web page, elements of the page can be served by third-parties (e.g. banner ad).

  You can configure browsers to not accept third-party cookies
- Supercookies – Top level domain cookies?! Blocked by browsers
- Zombie cookies – get recreated if they get deleted (by local code: Flash, JavaScript)

Cookies and European law



This site uses cookies to store information on your computer.

- European Union Directives

  Directives direct EU member countries to enact country level laws

  In 2002 launched the Directive and Privacy in Electronic Communications (DPEC)

  Article 5 paragraph 3 – a user needs to be informed of how information about them stored on their device will be used, and given the option to opt-out

  In 2009 changed the directive to require opt-in rather than opt-out

Image credits

All drawings and photos by Ben Gaucherin unless noted

Slide#	credit
2-6	https://i.guim.co.uk/img/static/sys-images/Media/Pix/pictures/2012/7/19/1342719614591/Viral-video-Sesame-Street-009.jpg?w=1200&q=85&auto=format&sharp=10&s=1476df7ec35559f339e4be56e0613894
7	https://www.drupal.org/files/project-images/cookie%20control%20v4.jpg

Internet application protocols

Session Initiation Protocol (SIP)

CSCI E 45b: The Cyber World – part B

Overview

- IETF Proposed Standard - RFC 3261
- Protocol to setup multimedia sessions
  - Audio, video
  - Used in Voice over IP (VoIP)
  - Application-layer signaling protocol
- SIP invitations include session descriptions

SIP - basics

- Call establishment and handling facets
  - User location - find out user's current location
  - User capabilities - find out media parameters to use
  - User availability - find out if user wants to participate
  - Call setup - establish call parameters at called & calling
  - Call handling - includes call transfer & call termination

SIP - basics, contd.

TAKE A SIP

- Can gateway with PSTN (telephone network)
- Works with Session Description Protocol (SDP)

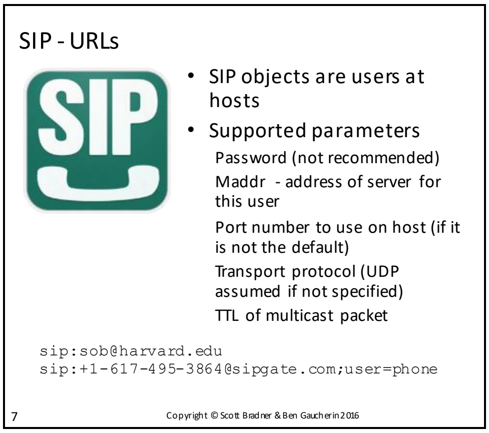
Session Description Protocol (SDP)

- Language to describe media sessions
- Includes:

session name and purpose
timing of session
media descriptions
format information
contact information
resource requirements

SDP example

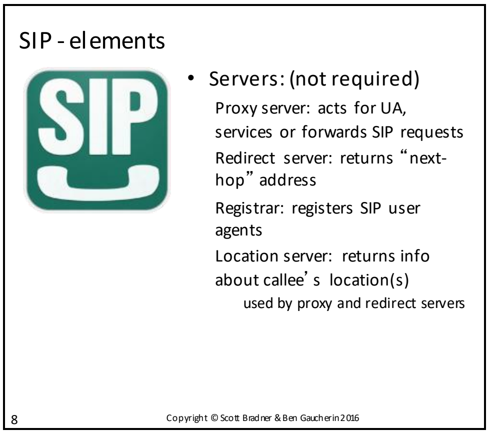
```
v=0
o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
e=mjh@isi.edu (Mark Handley)
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 31
m=application 32416 udp wb
a=orient:portrait
```



SIP - URLs

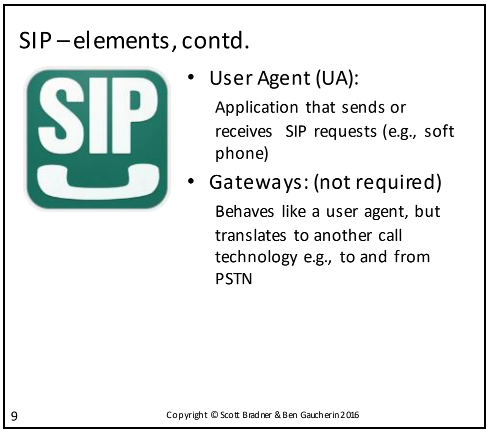
- SIP objects are users at hosts
- Supported parameters
  - Password (not recommended)
  - Maddr - address of server for this user
  - Port number to use on host (if it is not the default)
  - Transport protocol (UDP assumed if not specified)
  - TTL of multicast packet

```
sip:sob@harvard.edu
sip:+1-617-495-3864@sipgate.com;user=phone
```



SIP - elements

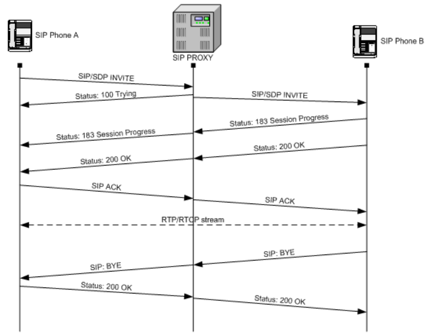
- Servers: (not required)
  - Proxy server: acts for UA, services or forwards SIP requests
  - Redirect server: returns "next-hop" address
  - Registrar: registers SIP user agents
  - Location server: returns info about callee's location(s)
    - used by proxy and redirect servers



SIP – elements, contd.

- User Agent (UA):
  - Application that sends or receives SIP requests (e.g., soft phone)
- Gateways: (not required)
  - Behaves like a user agent, but translates to another call technology e.g., to and from PSTN

SIP – high-level flow



10

SIP - messages

- Text-based using UTF-8 encoding
- Messages exchanged in RFC 822 (email) format
  - 1 or more headers
  - blank line
  - optional message body
- Headers
  - general-header, entity-header, request-header, response-header

11

SIP – request messages

- <Method> Request-URI SIP-Version
- Methods:
  - INVITE - invite participation in session
  - ACK - acknowledge INVITE
  - OPTIONS - negotiate capabilities
  - BYE - end session
  - CANCEL - cancel INVITE
  - REGISTER - bind URI address to a location
- example
  - INVITE sip:sob@harvard.edu SIP/2.0

12

SIP – other methods

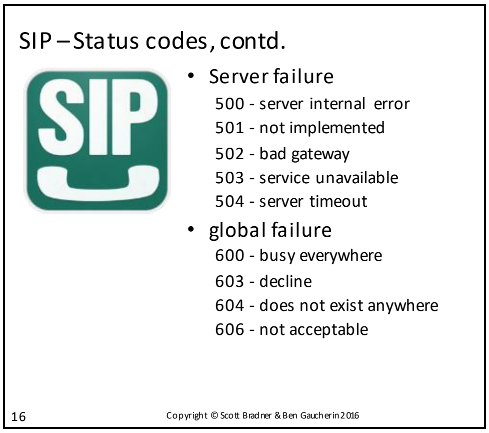
- INFO - mid-call information
- COMET - precondition met
- PRACK - provisional acknowledgement
- SUBSCRIBE - subscribe to event
- NOTIFY - notify subscribers
- REFER - call transfer

SIP – Status codes

- Informational
  - 100 - trying
  - 180 - ringing
  - 181 - call is being forwarded
  - 182 - queued
  - 183 - session progress
- Success
  - 200 - OK

SIP – Status codes, contd.

- Redirection
  - 300 - multiple choices
  - 301 - moved permanently
  - 302 - moved temporally
  - 305 - use proxy
  - 380 - alternative service possible
- Request failure
  - 400 - bad request
  - 401 - authorization required
  - 404 - not found
  - 407 - proxy authorization required
  - 486 - busy here



SIP – Status codes, contd.

- Server failure
  - 500 - server internal error
  - 501 - not implemented
  - 502 - bad gateway
  - 503 - service unavailable
  - 504 - server timeout
- global failure
  - 600 - busy everywhere
  - 603 - decline
  - 604 - does not exist anywhere
  - 606 - not acceptable

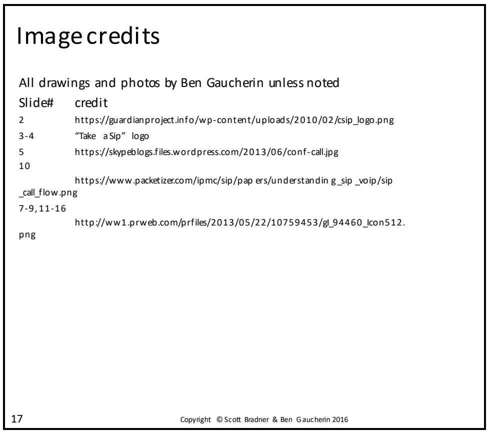


Image credits

All drawings and photos by Ben Gaucherin unless noted

Slide#	credit
2	https://guardianproject.info/wp-content/uploads/2010/02/csip_logo.png
3-4	"Take a Sip" logo
5	https://skypeblogs.files.wordpress.com/2013/06/conf-call.jpg
10	https://www.packetizer.com/ipmc/sip/papers/understanding_sip_voip/sip_call_flow.png
7-9, 11-16	http://ww1.prweb.com/prfiles/2013/05/22/10759453/gl_94460_Icon512.png

Internet application protocols

Secure SHell (SSH)

CSCI E 45b: The Cyber World – part B

History

SSH!
DON'T GIVE AWAY YOUR PASSWORD TO ANYONE WHO MIGHT BE LISTENING!
Always use SSH (Secure Shell) when connecting to Computer Science Department hosts, instead of telnet or rsh. For more information:
http://www.cs.umd.edu/faq/ssh.html

- SSH-1 developed by Tatu Ylönen in 1995 to address security issues of rlogin, Telnet, and rsh
- OpenSSH became the leading Open Source implementation of SSH

History, contd.

SSH!
DON'T GIVE AWAY YOUR PASSWORD TO ANYONE WHO MIGHT BE LISTENING!
Always use SSH (Secure Shell) when connecting to Computer Science Department hosts, instead of telnet or rsh. For more information:
http://www.cs.umd.edu/faq/ssh.html

- IETF Secsh workgroup developed SSH-2 in 2006 to address a number of design flaws in SSH-1
- SSH has associated utilities using the SSH protocol for specific functions: e.g. `scp` to securely copy files over SSH

SSH - basic use, and process

- SSH allows access to a shell (command line interface) on a remote machine over a secure connection
- It goes through three steps to establish a connection:
  - Host identification - Proving the remote host you are talking to is the one you think it is
  - Encryption – Secure the end-to-end connection
  - User authentication – Using username/password or key pairs
- This is a simple VPN for the user

SSH – simple tunnel

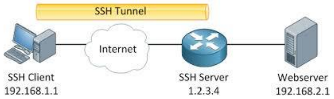
Wireless network
THE "INTERNETS"
Laptop
SSH Tunnel
Email
Email/web/ssh Server

- Another use of SSH is to create a simple tunnel through which a local port is connected to the same port on the remote machine
- This is a simple VPN for an application
- This is useful to secure a connection to legacy applications that do not have secure protocols

```
ssh ben@192.168.0.1 -L80:192.168.0.1:80
```

SSH – more complex use

- Bastion host
  - Term coined by Marcus Ranum
  - A host designed to withstand attacks, and generally run only one service
- Using SSH you establish a tunnel between your local machine and the bastion host, and can forward traffic to another machine on the local network of the bastion host



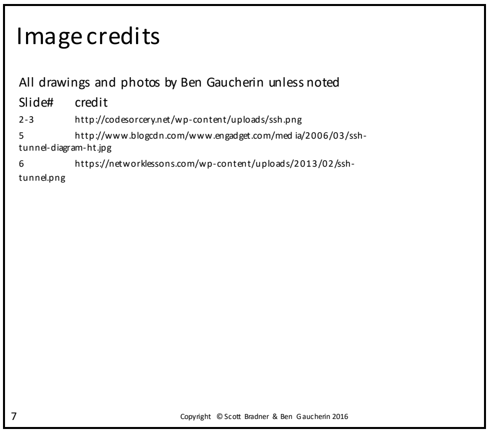


Image credits

All drawings and photos by Ben Gaucherin unless noted

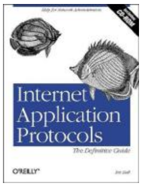
Slide#	credit
2-3	http://codesorcery.net/wp-content/uploads/ssh.png
5	http://www.blogcdn.com/www.engadget.com/med ia/2006/03/ssh-tunnel-diagram-ht.jpg
6	https://networklessons.com/wp-content/uploads/2013/02/ssh-tunnel.png

Internet application protocols

Conclusion

CSCI E 45b: The Cyber World – part B

In summary…



- A surprisingly small number of relatively simple protocols power the vast majority of what we do on the Internet today
- Remember, the Internet is a “stupid network”. All you need are end points (client and server) that communicate with each other.

Internet application protocols take the IP layer (and others below) for granted

In summary…

- Some of these protocols have gained a lot of popularity and are evolving to be generic information request/response protocols way beyond their original intent

e.g., HTTP and web services

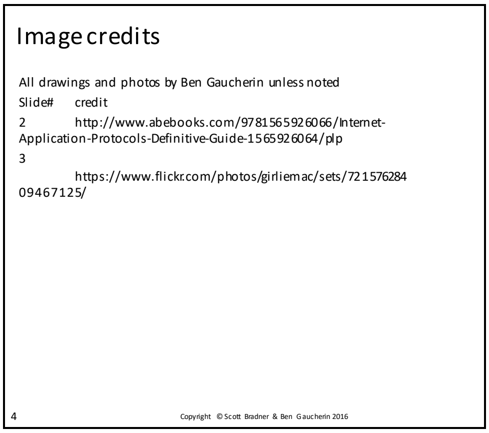


Image credits

All drawings and photos by Ben Gaucherin unless noted

Slide#	credit
2	http://www.abebooks.com/9781565926066/Internet-Application-Protocols-Definitive-Guide-1565926064/plp
3	https://www.flickr.com/photos/girliemac/sets/72157628409467125/