Networks and protocols

How computers exchange information over networks

Prerequisites

We assume that you have read the following prior to attending this workshop:

- Kernigan: Part III - Communications, Internet, pages 117-160
- Wiki:
  - http://wiki.juneday.se/mediawiki/index.php/ITIC:Networks_and_protocols
  - http://wiki.juneday.se/mediawiki/index.php/ITIC:HTML - in particular the videos:
    - HTTP Introduction (Full playlist) | HTTP Introduction - 01 | 02 | 03 | 04 | 05 | pdf
- Compendium (Swedish): *Ladda ned filer från webben*, pages 40-44

Objectives

Understand

- How computers are connected via networks
- How domain names are resolved to IP addresses
- What protocols are
- What data formats are and why we need them

Know:

- What an IP number is (including private and public networks)
- Some data formats such as XML, Json and CSV

Work in groups

Work in groups during this workshop. You may use existing groups or form new groups of 4-5 people.

Install a few applications

Make sure you have

- `jq`
- `curl`
- `dos2unix`
- `lwp-request (package libwww-perl)`

```
$ sudo apt-get update && sudo apt install jq curl dos2unix
libwww-perl
```

Names and numbers

- We’ll warm up with some investigations of your IP numbers and domain names.

Find out your local IP address

- Find you your local IP address (there are many ways to do that)
- Write down both IPv4 and IPv6 addresses
- Compare with your group mates
  - Do the addresses look similar?
  - How, or how not?

Find out your local IP address

- You can use
  - `ip address`
  - `ifconfig`
  - `hostname -I`
  - Settings dialog (GUI)

Find out your external IP

- Your external IP is the one servers on the internet see when you visit them
- Write this down and compare with your group mates
  - Is it similar to that of your mates?
  - Why/why not, do you think this is?
- How does it differ from your local IP?

One simple way to find out your external IP

```
$ curl -s 'https://api.ipify.org?format=json' | jq .
{
  "ip": "129.16.30.99"
}
# or:

$ curl -s 'https://api.ipify.org'
129.16.30.99
```

What is the name of that computer?

- If your external IP resolves to a domain name, then you can ask a DNS to resolve it for you (example from Rikard’s computer):

```
$ host 129.16.30.99
99.30.16.129.in-addr.arpa domain name pointer ext-1-100.eduroam.chalmers.se.

$ nslookup 129.16.30.99
Server:      127.0.1.1
Address: 127.0.1.1#53
Non-authoritative answer:
99.30.16.129.in-addr.arpa name = ext-1-100.eduroam.chalmers.se.
```

Play with HTML

- Next we are going to download a web page from the university and edit it to include yourself and view the result locally in a browser (without network)
- Purpose it to show you how you can download files over HTTP
- Expose you to some HTML code
- Use an editor

Download a web page from gu.se

- Download this page https://ait.gu.se/kontaktaoss/journalister
- Save it as experten.html

You may use `wget` with the URL as argument and `-O experten.html` at the end.

Before you start, create a directory for this exercise.

Change the file from Windows to Unix

- The file might have been created on Windows and look funny on GNU/Linux
- Windows uses a different line-ending indicator (CRLF) from that of Unix (LF)
- Run the command `dos2unix experten.html`
- Next, remove empty lines (there are tons of them for some reason)
- `grep -v '^$' experten.html > experten2.html`
- You will now work with experten2.html or change the name to whatever you feel

Edit the file

- Open the file in an editor from the terminal
- Practice using TAB completion
- Use nano, gedit or an editor of your own choice
- Search for “Artificiell Intelligens” (no idea why it is capitalized)
- Find the <li> (list item) element containing this entry
- Make it stand on its own line

Edit the file

```
  <li><strong>Artificiell Intelligens (AI)</strong><br /><a
href="https://ait.gu.se/om-institutionen/avdelningen-for-kog
nition-och-kommunikation/personal?userId=xlowro">Rob
Lowe</a>: 031-786 2791<br /><a
href="https://ait.gu.se/om-institutionen/avdelningen-for-inf
ormatik/personal?userId=xivajo">Jonas Ivarsson: </a>031-786
2473<br /> </li>
```

Duplicate the line (cut it and paste it twice)

```
  <li><strong>Artificiell Intelligens (AI)</strong><br /><a
href="https://ait.gu.se/om-institutionen/avdelningen-for-kognition-o
ch-kommunikation/personal?userId=xlowro">Rob Lowe</a>: 031-786
2791<br /><a
href="https://ait.gu.se/om-institutionen/avdelningen-for-informatik/
personal?userId=xivajo">Jonas Ivarsson: </a>031-786 2473<br
/> </li>
  <li><strong>Artificiell Intelligens (AI)</strong><br /><a
href="https://ait.gu.se/om-institutionen/avdelningen-for-kognition-o
ch-kommunikation/personal?userId=xlowro">Rob Lowe</a>: 031-786
2791<br /><a
href="https://ait.gu.se/om-institutionen/avdelningen-for-informatik/
personal?userId=xivajo">Jonas Ivarsson: </a>031-786 2473<br
/> </li>
```

Edit your copied line (the first one)

- Put your own name and expert topic (or a fake name if that’s your choice)
- Make the link around the name go to some actual page (your home page or some other page)
- Make the phone number a fake number (or your own) - you are the only one who can see this web page, since it is on your own computer
- Save the file

Open the file in a browser

- Rikard uses Google Chrome, so on his computer:

```
$ google-chrome experten2.html
```

- Find out how to do the same if you use another browser

File | /home/rikard/itid/experten/experten2.html

GÖTEBORGS UNIVERSITET

English | Lyssna

TILLÄMPAD INFORMATIONSTEKNOLOGI

FORSKNING UTBILDNING SAMVERKAN AI OM INSTITUTIONEN PERSONAL FÖR PERSONAL STUDEN

Göteborgs universitet / Tillämpad IT / Kontakta oss / Hitta experten

Kontakta oss

Hitta experten

Studieadministration

Studievägledning

Hitta experten!

Hos oss hittar du forskare med svar på frågor som rör samhällets digitalisering. Om du söker ett ämne eller fråga som inte finns listad nedan - kontakta pressansvarige Peter Larsson.

- **Rock'n'roll lifestyle**
  Rikard Fröberg: 031-90 51 06

A few words on the HTML you edited

```
<ul> <!-- unnumbered list -->
  <li> <!-- list item -->
    <strong> <!-- bold font face -->
      Space exploration
    </strong><!-- bold font face ends →
    <br /><!-- linebreak →
    <a href="http://wiki.juneday.se">
      Rikard Fröberg</a> <!-- Hyperlink with text -->
  </li><!-- list item ends -->
</ul><!-- unnumbered list ends -->
```

A few words on the HTML you edited

```
<ul>
  <li>
    <strong>Space exploration</strong><br />
    <a href="http://wiki.juneday.se">Rikard Fröberg</a>
  </li>
</ul>
```

How was the file downloaded?

- You used a client capable of talking HTTP to GU's web server
- The client sent a request (with some headers and a method and a resource)
- The server (web server running at ait.gu.se) sent a response with some headers and the resource (in this case, the web page)

HTTP GET request

Use telnet to send this (end with two lines):

```
GET /kontaktaoss/journalister HTTP/1.0
Host: ait
```

HTTP response

```
HTTP/1.1 301 Moved Permanently
Server: nginx/1.12.2
Date: Thu, 01 Aug 2019 09:10:17 GMT
Content-Type: text/html
Content-Length: 185
Connection: close
Location: https://ait/kontaktaoss/journalister

<html>
<head><title>301 Moved Permanently</title></head>
<body bgcolor="white">
<center><h1>301 Moved Permanently</h1></center>
<hr><center>nginx/1.12.2</center>
</body>
</html>
```

301 Moved Permanently

- Instructs a client to go fish somewhere else (The Location header tells you where)
- Your browser follows redirects for you
- Telnet doesn’t (Telnet doesn’t speak HTTP)
- curl and wget can follow redirects (curl needs to be asked, wget just does it)
- This is common - you enter a url and are redirected

200 OK

- This is what we want - It means that the server says it can fulfil the request

404 Not Found

- Means the server couldn’t find the requested resource (file)

```
$ telnet ftp.sunet.se 80
Trying 2001:6b0:19::165...
Connected to sunet.ftp.acc.umu.se.
Escape character is '^]'.
GET / HTTP/1.1
Host: ftp

HTTP/1.1 200 OK
Date: Thu, 01 Aug 2019 09:21:17 GMT
Server: Apache/2.4.39 (Unix)
Last-Modified: Tue, 12 Mar 2019 15:35:37 GMT
Cache-Control: max-age=300
Expires: Thu, 01 Aug 2019 09:25:54 GMT
Age: 22
Content-Length: 4641
Content-Type: text/html;charset=UTF-8

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>
 <head>
  <title>Index of /</title>..... etc
```

Requesting headers

- You can use `lwp-request -m HEAD` (on GNU/Linux you can use the alias `HEAD`) to only get the headers from a server:

```
$ lwp-request -m HEAD http://ftp.sunet.se/
200 OK
Cache-Control: max-age=300
Connection: close
Date: Thu, 01 Aug 2019 09:25:02 GMT
Age: 151
Server: Apache/2.4.39 (Unix)
Content-Type: text/html;charset=UTF-8
Expires: Thu, 01 Aug 2019 09:27:30 GMT
Last-Modified: Tue, 12 Mar 2019 15:35:37 GMT
Client-Date: Thu, 01 Aug 2019 09:25:02 GMT
Client-Peer: 2001:6b0:19::173:80
Client-Response-Num: 1
```

Group task

- Find out what web server is used by the following domains:
- http://snarxiv.org/
- http://www.elsewhere.org/
- https://ait.gu.se/
- http://www.gu.se/
- https://pts.se
- https://opensource.org/
- http://gnu.org
- http://chalmers.se
  - Hint: Use `HEAD` (or `lwp-request -m HEAD`) and `grep` to find the correct header line (starts with **`Server:`**)

Port numbers

- Did you notice the 80 at the end of the telnet command?
- 80 is the standard port for HTTP
- You can add a port to a url directly after the domain name (works e.g. in your browser): http://ftp.sunet.se:80/some/resource
- Since 80 is default it is not necessary
- HTTPS uses 443
- Do you remember the apartment number analogy?

Data formats

- Next, we will use some data formats commonly used
- Data can be encoded as plain text but often we use some format to markup the data, in order to give the text meaning
- If you get the following data from a computer, what does it mean?
  `86785500`
- A phone number? In which country? An order number? Something else?
- If you instead got the following text:
  `<phone country_code="SE">86785500</phone>`
  things might be clearer...

Images can actually be encoded as text

- Download https://raw.githubusercontent.com/progund/intro-it/master/network-protocols-data/workshop/pic.html
- Open it in your browser
- View source (in Chrome: Ctrl-U or use right-click → view source)

Json

- This section aims to get you familiar with Json
- You will also see that Json is a common data format for Open APIs

What can a barcode tell us?

- The following URL is created from a barcode of a product we had in our office: https://world.openfoodfacts.org/api/v0/product/4009900484220.json
- Use the following command line to see it formatted by jq in your terminal:
  ```
  $ GET https://world.openfoodfacts.org/api/v0/product/4009900484220.json | jq .
  ```
- Use grep to find lines containing “image_ingredients_url”
- Open the URL in your browser
- What product was it?
- Group work: Find a food product bar code and see if you can get the corresponding Json file from it (use the number as part of the URL)

jq can help extracting parts of Json

```
$ GET https://world.openfoodfacts.org/api/v0/product/4009900484220.json | jq
."product"."image_ingredients_url"
"https://static.openfoodfacts.org/images/products/400/990/048/4220/ingredients
_en.7.400.jpg"

# The jq path was:
  ."product"."image_ingredients_url"
# Other interesing paths you can use for your products:
."product"."nutriments"."sugars"
."product"."allergens_tags"
."product"."nutriments"."energy" (I think kJ/100 gr)
```

You are hungry and your fridge almost empty

You find:

- vodka
- celery
- hot sauce
- tomato juice
- salt
- pepper
- lemon juice

What do you do?

You are hungry and your fridge almost empty

You use Bash and the Internet, of course:

```
GET
'http://www.recipepuppy.com/api/?i=vodka,celery,hot+sauce,tomato+juice,salt,black+pepper,le
mon+juice&p=3'|jq ."results"[]."title"
"Bloody Mary Soup Recipe"
"Emeril's Bloody Mary Mix"
"Tomato Mocktail"
"Bloody Mary With Spicy Ice Cubes"
"Cajun Bloody Mary"
"bloody mary soup"
"Coronado Gazpacho"
"Cajun Bloody Mary Recipe"
"Gazpacho Recipe"
"Bloody Mary"
```

You are hungry and your fridge almost empty

You use Bash and the Internet, of course:

```
$ GET
'http://www.recipepuppy.com/api/?i=vodka,celery,hot+sauce,tomato+juice,salt,black+pepper,le
mon+juice&p=3'|jq ."results"[9]."ingredients"
"vodka, tomato juice, lemon juice, worcestershire sauce, hot sauce, salt"

$ GET
'http://www.recipepuppy.com/api/?i=vodka,celery,hot+sauce,tomato+juice,salt,black+pepper,le
mon+juice&p=3'|jq '."results"[9]|."title" + ": " + ."ingredients"'
"Bloody Mary: vodka, tomato juice, lemon juice, worcestershire sauce, hot sauce, salt"
```

Now you do it!

- Group work: What recipes can you find if you have:
- bacon, pasta, parmesan cheese, butter, black pepper, eggs
- Hint: use + as space in words with multiple letters: black+pepper

Sometimes lwp-request is not allowed

- Then switch to curl and spoof your user-agent
- When has Johanna her name day in Sweden?

```
$ curl -s -A 'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/73.0.3683.103 Safari/537.36'
'https://api.abalin.net/get/getdate?name=Johanna&calendar=se'|jq .
{
  "calendar": "se",
  "results": [
    {
      "day": 21,
      "month": 7,
      "name": "Johanna"
    }
  ]
}
```

Spoofing user-agent

```
# flag -A 'user-agent string':

curl -A 'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36'

# user-agent (example):
Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36
```

CSV

- Download this file: https://people.sc.fsu.edu/~jburkardt/data/csv/grades.csv
- Open it in a spreadsheet application (like LibreOffice Calc or Excel or Drive)
- There’s an error in the file. Can you find and fix it?

Summary

- Private IP - on private network
- Public IP - shared by nodes in the private network
- DNS - allows you (and programs) to look up IP from name and vice versa
- HTML - looks cryptic but isn’t that hard to lean (at least the basics)
- Windows files have different line-endings
- HTTP - we looked at request methods GET and HEAD and responses
- Status codes 301, 200, 404
- Port numbers - we used 80 for HTTP with telnet
- Base64 encoded image
- Json: we used lwp-request, curl and jq to get data as Json
- CSV - we looked at a (faulty) CSV file