Computer Networks

How do computers communicate?
Computer Networks

HOW DO COMPUTERS COMMUNICATE?

Useful as individual computers are for running programs, such as games, calendars and spreadsheets, to perform calculations and help manage information, it has really been through connecting computers together to form networks, and particularly through the internet, a network of networks, that they have had the most immediate impact on our lives. Consider how limited our use of technology in school would be if we had no access to the local network or the internet. Consider how frustrating many find it when, even temporarily, we have no data signal for smartphones or WiFi for a tablet or laptop.

The internet has made possible communication and collaboration with a diversity and immediacy never before possible, and yet, perhaps like writing, printing and the telephone before it, it’s something that most of us take for granted, and possibly have little understanding of. The computing curriculum sets out to change this: alongside developing pupils’ computational thinking through practical programming, it includes requirements that primary pupils be taught to ‘understand computer networks, including the internet [and] how they can provide multiple services, such as the world wide web’ and to ‘use search technologies effectively, appreciate how results are selected and ranked, and be discerning in evaluating digital content’ (Department for Education [DfE], 2013). At Key Stage 3, the requirements include that pupils should be taught how the hardware and software components that make up computer systems ‘communicate with one another and with other systems’.

How does the Internet work?

The internet is a physical thing: it is the cables, fibre, transmitters, receivers, switches, routers (and all the rest of the hardware) that connects computers, or networks of computers, to one another.

The internet has been designed to do one job: to transport data from one computer to another. This information might be an email, the content of a web page or the audio and video for a video call.

The data that travels via the internet is digital: this means it is expressed as numbers. All information on the internet is expressed this way including text, images and audio. These numbers are communicated using binary code, which is made up of 1s and 0s, using on/off (or low and high) electrical or optical signals (see page xx). Binary code is similar to the Morse code used for the telegraph in Victorian times, but it’s much, much faster. A good telegraph operator could work at maybe 70 characters (letters) a minute, but even a basic school network can pass data at 100 million on/off pulses a second, enough for some 750 million characters per minute. One transatlantic fibre connection has the capacity for up to 24 trillion characters per minute.

Digitised information needs to be broken down into small chunks by the computer before it can be sent efficiently. These smaller chunks of data are known as ‘packets’.

The small packets can be passed quickly through the internet to the receiving computer where they are re-assembled into the original data. The process happens so quickly that high definition video can be watched this way, normally without any glitches.

The packets don’t all have to travel the same way through the internet: they can take any route from sender to recipient. However, there is generally a most efficient route, which all the packets would take (Figure 4.1).
It is perhaps easier to understand how the internet works now by looking at a picture of how it worked in 1969 when it started (Figure 4.2):

![Figure 4.2](image)

Here you see the internet made up of just four routers: UCLA, SRI, UCSB and UTAH. Each router is a piece of hardware that passes packets of data from the computers they are connected to (in the case of UTAH, PDP10, in the case of UCLA, SIGMA 7), and perhaps any terminals connected to those computers, to any of the other three computers and their terminals.

So if you were using the PDP10 computer at the University of UTAH and sent a message to someone at UCLA, your message would be passed first to your router at UTAH, then on to the router at Stanford Research Institute (SRI), then (normally) to UCLA’s router, where it would be passed on to whichever recipient it was intended for on their SIGMA 7.

The internet is obviously much, much bigger than this example. In real life, the journey of a packet of data from your home computer to one of Microsoft’s server farms might look something like this:

- your home WiFi access point;
- your home switch and router (usually all in the same black box);
- switches in your nearest BT green cabinet;
- more switches in your local telephone exchange;
- London internet exchange;
- routers near Porthcurno in Cornwall;
- fibre optics under the Atlantic;
- further switches and routers in the USA until Microsoft’s internet connection at whichever of its data centres you are communicating with.

When you type a URL (such as www.bbc.co.uk or www.computingatschool.org.uk) into your browser you send a packet of data requesting the content of these pages to be returned to you. But before this can happen, the domain name first needs to be converted into numbers. This is the job of the Domain Name Service (DNS), which converts these familiar web addresses into numbers known as Internet Protocol (IP) addresses. The DNS itself uses the internet to look up (in the equivalent of huge phone books) the numeric address corresponding to the domain names, but it keeps a local record (cache) of these, so that the next time the domain name is requested, the IP address can be returned more quickly.

Each packet has a destination IP address on it. With it the router can easily look up which way to pass the packet on.

---

1 From [www.computerhistory.org/internet_history/](http://www.computerhistory.org/internet_history/)
Who can see the data we transmit?

There’s nothing to stop routers from looking at the data in the packet before they pass it on (just as there was nothing to stop telegraph clerks reading the messages they passed on in Morse code).

To be able to send information, such as passwords or bank account details, secretly via the internet, it’s important to encrypt the data first. This happens automatically when using the ‘https’ version of websites. In these situations, you will see a little green padlock displayed in your browser’s address bar. The data are decrypted when they reach their destination – see pages xx–yy for more on cryptography.

Classroom activity ideas

- Ask pupils to draw a picture of the internet. This will allow you to spot any misconceptions they have, and provide an opportunity for pupils to share their understanding.

- Carry out this ‘unplugged’ activity to model how the internet passes packets of data.
  - Organise all but four of your pupils into groups.
  - Tell the pupils to choose one pupil in their group to be the ‘group router’. The rest of the group will be ‘computers’.
  - Ask the remaining four pupils to take on the role of ‘internet routers’, which connect the group routers together.
  - Give each ‘computer’ a numerical address, comprising a group number and a computer number (for example 1.1, 1.2, 1.3; 2.1, 2.2, 2.3, and so on; Figure 4.3).
  - Ask each ‘computer’ to write a short message to another ‘computer’ in a different group, splitting their message over three different slips of paper and marking their slips ‘1 of 3’, ‘2 of 3’ and ‘3 of 3’. Tell them to write their numerical address and the numerical address of the recipient, for example ‘To: 2.2; From 3.4; 2 of 3’. This is the ‘packet header’.

- Investigate the physical infrastructure of the school network. Tell the pupils to walk from their laptop to the local WiFi point or to follow the network cable from the computer to the classroom switch. Next, walk together to the school’s main network switch, firewall and router. If you can, then walk down to the nearest BT green cabinet, and perhaps to your local telephone exchange, depending on how close this is to you.

- Explore the steps on the journey of a packet using the ‘tracert’ command at the Windows command prompt, if you have access to this. Also see the Visual traceroute reference in Further resources.

- Ask your school network manager to talk pupils through how the school network connects their computers to the rest of the internet.

Further resources


What can you do with the Internet?

One way to think of the internet is as the train network, efficiently routing trains of all kinds from one point to another, irrespective of what those trains contain: some will have passengers, others freight, others are perhaps maintenance stock. Similarly the infrastructure of the internet can be used for lots of different things. At present, we are most familiar with the web as the main application of the internet, but the internet pre-dates the web by a couple of decades and there are many who think we will be using the internet, or something very like it, long after the web becomes a historical curiosity.

The services which run on computer networks, including the internet, fall into roughly two groups (Figure 4.4):

(1) client–server: one computer (the client) accesses services or content running or stored on another, typically larger, computer (the server);

(2) peer-to-peer: two computers communicate directly as equals, passing data directly to and from each other.

The World Wide Web (see page 118) fits into the client–server model, but so do lots of other services which use computer networks and the internet as a means of communicating.

A school network will often have one or more computers acting as servers, responding to requests from the desktop, laptop and tablet computers which act as clients. On a local area network (LAN) like this, the servers might provide: central storage and backup for files, access to documents, and so on, from any computer on the network, a management information system (such as SIMS, Student Information Management System), local email accounts, access to printers, username and password authentication, filtering and logging of access to the web and even locally-stored copies of frequently visited web pages.

Email is a good example of a client–server system using the internet (although many people’s experience of email is as webmail accessed through a browser like Internet Explorer or Chrome). The journey of an email might be something like this:
Alice opens up Outlook and starts typing in her email to Bob. She includes Bob’s email address, bob@builders.com, in the ‘To’ line of the email and clicks ‘send’.

The email is transmitted via the internet (or the local network) to her outgoing mail server. If the email is intended for another domain (builders.com here) rather than Alice’s own (lookingglass.org) then Exchange will forward the email as packets of data via the internet, which routes these through to the incoming mail server for builders.com as discussed above.

The inbound mail server at builders.com (again perhaps running Exchange) re-assembles the message from the packets of data, accepts this and stores this ready for Bob to collect.

Later on, Bob’s email client (perhaps also Outlook) connects to his mail server and asks if there are any messages for him. The one from Alice gets transmitted to Bob’s computer via the local network or the internet, where Bob can read it in his email software.

Although it might look to Alice and Bob as though they are communicating directly with each other, all their emails are going via the outbound and inbound mail servers. Notice that the contents of their emails are not encrypted, so the organisations running the two mail servers can read the contents of these messages if they wish.

Not all communication on the internet uses a client–server model. For example, peer-to-peer communication is a model used for Skype and a number of other video conferencing or voice over internet systems. Although Skype uses a server to maintain a list of logged-in users and the IP address of their computers, when a call is connected the packets of data that make up the digitised video and audio for the call are routed directly through the internet between the two parties.

Some online gaming websites use a similar peer-to-peer system, as does BitTorrent (Cohen, 2003), a protocol which allows large files to be shared between many computers by allowing direct peer-to-peer connections, and blockchain (Nakamoto, 2008), a distributed ledger system for transactions in cryptographically-generated (‘mined’) currency. Because peer-to-peer connections are harder for large organisations to monitor, they are favoured by those using the internet for criminal purposes, for example the use of the BitTorrent protocol for illegally sharing copyrighted material or the blockchain-based Bitcoin for purchasing illegal goods.
What is the World Wide Web?

In 1989, British computer scientist Tim Berners-Lee decided to combine the capabilities of the internet with the functions of hypertext (documents that include hyperlinks that allow connections to be made between different files; see Figure 4.5) to manage information systems at CERN where he was working (Berners-Lee, 1989).

Berners-Lee developed a specification for how an internet-based version of hypertext would work and then wrote the software for the first web servers and web browsers. The result was the World Wide Web.

The internet is about connecting computers together, but the World Wide Web is about the connections between documents (Figure 4.6). When you click on a web link, another web page is requested from (typically) a different web server somewhere else on the internet.

The content of this web page is then delivered to your web browser.

What standards does the World Wide Web use?

To ensure that all computers could communicate with one another, Berners-Lee developed a set of standards (called protocols) for the web. Versions of these are all still used today.

HTTP (HyperText Transfer Protocol)

This is the process that computers use to request and transfer hypertext to one another.

The web is a client–server system: we use a web browser on our computer to request a web page from one of the many, many web servers connected to the internet. The request travels as a packet of data via switches and routers until it reaches the intended web server. The server responds by sending back the content of the page, together with any images and formatting instructions and mini programs (typically in JavaScript) needed for the page. If the page isn’t there, it sends back a ‘404: Not found’ error message – sometimes you will see other error messages too.

Remember that the internet doesn’t encrypt packets of data: there’s another version of HTTP, called HTTPS, where the request for a page, the contents of the page and any information entered into a form (such as a password) are sent over the internet in an encrypted form. This encryption can sometimes be bypassed by network managers and government agencies.
URL (Uniform Resource Locator)

URLs are the precise location on the web where web pages or their components are stored. It’s what you type in to your browser’s address bar to request a page.

Each bit of a URL means something. Let’s look at the URL of one of the first web pages – Berners-Lee’s home page for the World Wide Web project itself – to work out what each bit means:

http://info.cern.ch/hypertext/WWW/TheProject.html

- http: this is the protocol we are using to request hypertext and the content that comes back – see above;
- :// is just punctuation – Berners-Lee now thinks it would have been better if he’d skipped the // bit!;
- info is the name of the web server we are connecting to. Often this will be www these days, or this is just omitted as the main web server for the organisation will be assumed;
- cern is the name of the organisation, in this case the European Centre for Nuclear Research;
- ch is an abbreviation for the country where the organisation has registered their domain name, in this case Switzerland. Some countries also show what sort of organisation it is registered as, for example .co.uk for a commercial site and .sch.uk for a school site in the UK. If no country is shown, then it will be registered in the USA: .com for commercial sites, .edu for university sites, and so on;
- /hypertext is a directory (folder) on the web server;
- /WWW is a directory inside the /hypertext directory on the web server;
- TheProject is the name of the actual file we are requesting, in this case a web page about the World Wide Web project. Sometimes you don’t see a file name at the end of a URL, in which case the web server will send back the default file for the directory, often an index page such as index.html;
- .html is the file extension, which shows what format the page is written in, in this case HTML. This is like .doc or .docx for a Word file or .jpg or .jpeg for an image.

Although it is often convenient to use search engines like Google or Bing to find pages rather than typing in URLs, the URL is a good way to check that you’re connecting to the web server you think you are (rather than a spoof website). URLs are also needed when acknowledging sources of information and for creating links between pages (and so building more of the connections that make the web so useful).

HTML (HyperText Mark-up Language)

HTML is the computer language (code) in which the content and structure of a web page are described or ‘marked up’.

The content of web pages is stored in HTML format on web servers. Creating a web page involves writing (or getting a computer to generate) the HTML that describes the page. HTML can be read, and written, by humans as well as computers. You can view the HTML source code for any web page using tools built into your web browser. (There’s a menu command to do this, or you can press ‘ctrl-u’ in Internet Explorer.)

These days, the HTML for a web page might not be stored as a file on the web server: in content management systems, when a page is requested it will be generated automatically using a database of content, a template and some programs running on the web server, perhaps written in Python or PHP. For example, every time you visit www.bbc.co.uk/newsround/ the page will be generated using the latest news in the database.

More recently, a couple of other languages have come to play an important part in developing for the web.

CSS (Cascading Style Sheets)

CSS provides formatting information alongside the content and structure of HTML, allowing designers and developers to specify exactly how the content of the page should be displayed in the web browser on a computer, tablet, smartphone or printer.

JavaScript

JavaScript is a programming language that can be interpreted by the web browser itself, allowing interaction with the content of a page to be handled by the user’s computer (the client) rather than on the server itself. The web-based version of Office 365 relies heavily on JavaScript.
What’s the most amazing thing about the web?

The amazing thing about the web isn’t really these technologies though. It’s that, from its early days as the preserve of academic scientists, so many organisations and individuals have connected their own web servers to the internet and added their own content to the web. In part this was because Berners-Lee created a system that was accessible, scalable and extensible, capturing the imagination of many. But it’s also because he and CERN gave it to the world for free – the standards and the technology were entirely open, without any central authority or commercial company licensing or charging for their use.

Classroom activity ideas

- Encourage pupils to look at the different parts of the URLs for the web pages they visit, asking them to explain what each part of the URL means. Make a display showing the different parts of some interesting or common URLs.
- Ask pupils to talk to their parents, grandparents or carers about the difference the World Wide Web has made in their lives.
- Tell pupils to keep a diary of the different ways they use the web over a week.
- It’s not too tricky to set up a webserver in school, although providing access to this from the rest of the internet may be harder. With access to a webserver, either on the school network or via the internet, pupils could create their own webpages either in HTML or using a content management system for others to view. They could install a number of open source applications such as Moodle or Wordpress, configuring these as they wish. They might also watch and analyse the data logged by the webserver as it responds to page requests across the network.³

Further resources

BBC Bitesize (n.d.) What is the world wide web? Available from www.bbc.co.uk/guides/z2nb9j27


How do you make a web page?

There are plenty of tools available for you and your pupils to create your own content for the web.

Your school’s learning platform or Virtual Learning Environment (VLE) provides one way to get content online, as do blogging platforms like WordPress. These platforms usually include a ‘WYSIWYG’ (what you see is what you get) editor. This makes writing content for the web similar to using Microsoft Word, with a range of formatting controls built in. In most of these editors, you can swap into code (or source view), seeing and editing the HTML itself. This can be a good introduction to working directly in HTML, as you can always swap back to the WYSIWYG view to see the effects of editing the code.

Giving pupils some experience of writing content for the web through editing HTML ‘by hand’ is well worth doing although it isn’t, strictly speaking, programming. It adds to their understanding of networks including the internet that the national

³ GitHub offers free hosting of static webpages: https://pages.github.com/
curriculum (at Key Stage 2) expects, and is one more way of using software on a range of devices to create content. It is also a good way to get pupils used to working in a formal, text-based computer language. As with other text-based languages, working in HTML helps reinforce the importance of spelling, punctuation and grammar: mistakes in the mark-up of the page usually become quite apparent in the way the browser displays the page.

Many pupils are likely to find these skills useful in the long term too, both at secondary school and beyond: developing content for the web is part of many jobs, teaching included.

**What does HTML look like?**

Let's compare the HTML code for a simple web page and the page itself.

```html
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>A simple webpage</title>
</head>
<body>
  <h1>Origins of the Web</h1>
  <p>Tim Berners-Lee started working on the world-wide web project in 1989.</p>
  <p>He was working at CERN in Switzerland at the time.</p>
  <img src="http://www.w3.org/Press/Stock/Berners-Lee/2001-europaeum-eighth.jpg">
</body>
</html>
```

**Figure 4.7**

Can you see where the content for the page (Figure 4.7) comes from in the code? Can you see what effect some of the HTML tags (the bits in the `<...>` angle brackets like `<h1> and `<p>`) have on how the content is structured?

Notice how most of the tags come in matched pairs, for example:

- `<html>` and ending `</html>` for the whole page;
- `<head>` to `</head>` for the information about the page, such as its character set and title;
- `<body>` to `</body>` for the content of the page;
- `<h1>` to `</h1>` around the main heading for the page;
- `<p>` to `</p>` around each paragraph.

Compare the underlined link in the web page with the corresponding code. In the code, `<a>` to `</a>` show where the link should be and `href="http://home.web.cern.ch/"` inside the `<a>` tag detail where the link should point to.

An image is inserted from elsewhere on the web, using a single `<img>` tag, this time without a matched closing tag, and again giving the location of the image using `src="http://www.w3.org/Press/Stock/Berners-Lee/2001-europaeum-eighth.jpg"` inside the `<img>` tag.
How do I get started with HTML?

Mozilla’s Thimble tool for creating websites (available at: https://thimble.webmaker.org/) makes it easy to get started with coding in HTML, as it displays the source code alongside the resulting web page, as does trinket.io in HTML rather than Python mode.

Rather than starting from a blank page, pupils can try editing other web pages, exploring the structure and HTML code of these pages and seeing what effect changing the code has on how the page is displayed in the browser.

On Internet Explorer or Chrome, you can use the Developer Tools (hit F12 or launch via the menu) to view and edit the source code (the HTML code which describes the content and structure) for a page. Alternatively, you can install Mozilla’s X-Ray Goggles as an active bookmarklet (see Further resources) to remix and share edited web pages.

Further resources


Playto (n.d.) App design basics. Learn to code using HTML and CSS. Available from https://learn.playto.io/html-css/lesson/0


Thimble (n.d.) Available from https://thimble.webmaker.org/

W3schools.com (n.d.) Tutorials on a wide range of computer languages. Available from www.w3schools.com/

Classroom activity ideas

● When using their learning platform, VLE or class blog encourage pupils to swap from the normal WYSIWYG (what you see is what you get) mode of the built-in editor into the code, source or HTML mode and try writing their post or page in that. Remind them that they can swap back and forth to see how the code relates to the page that’s displayed. Give pupils a list of some common HTML tags to try out for themselves.

● Set pupils the challenge of making a parody of a web page by using either the Developer Tools in Internet Explorer or X-Ray Goggles to edit the code for the page. It’s wise to decide some ground rules for this activity in advance. Show pupils how easily a spoof page can be created this way, and explain why it’s so important to check the address of the page they are visiting to confirm it is authentic rather than merely one which looks convincing.
How does a search engine work?

Search engines like Google and Bing have transformed the way we use the web. Instead of having to remember URLs for the pages we want or following the links from one page to another, we can normally rely on these web-based programs to give us the most relevant results for our query.

Given how much we use search engines, it’s important to use them effectively and efficiently, to show some discernment in deciding how far a particular page can be trusted, and to have some grasp of the algorithms that underpin them.

In order for Google or Bing to be able to respond to a search query, they use their index of the web. A search engine builds its index by using specially written programs called ‘web crawlers’. The web crawlers create a huge copy of the publicly accessible bits of the web (called a cache) which is stored on the search engine’s servers.

When a new or updated copy of a web page is added to the cache, an entry for the page will be added to, or updated in, the search engine’s index of the web for each of the words on the page (typically ignoring small, common words like ‘and’, ‘the’ and so on). The web crawlers continue to build and update the cache by following all the hyperlinks in the page, requesting and making copies of those pages too, adding or updating index entries for them and following the links on those pages too. And so on.

So when we type in a keyword such as ‘dog’ into a search engine, it consults the index and returns a list of all the web pages on which that keyword appears. Typing in several keywords, for example ‘dog’ and ‘bowl’ would only return pages with both of these keywords, which helps to narrow down the set of results.

How are search results ranked?

The really clever bit about web searches is not the list of results but the rank order the results are put into. How do the search engine algorithms decide what to put top of the list?

Google’s founders, Larry Page and Sergei Brin, recognised that the key to determining how relevant a particular result was likely to lie in the links between other pages and the result. They realised that a high-quality page is a page that has lots of links pointing to it from other web pages, particularly if they too were high-quality results (Page et al., 1999). This is shown in Figure 4.8, where the larger the circle, the higher the quality of the web page.

The cached and indexed copy of the (publicly accessible) web on the servers of search engines also includes the links between them. This allows Page and Brin’s PageRank algorithm to work out which pages are considered the highest quality to other web developers (as they add links to those into their own content). Thus, for many queries the Wikipedia entry will often be at the top of, or at least high up, the results list, not because of its accuracy or authority, or even because people click on this more than other results, but because lots of the other high quality search results link to it.

The actual algorithms that search engines use can be very complicated and are frequently tweaked to keep one step ahead of the ‘search engine optimisation’ (SEO) industry that tries to improve
the ranking for its clients’ pages. These days, the ranking of results is typically personalised: based on location, the history of what the user’s searched for and clicked on before, and close on 200 other factors or ‘signals’.

When teaching pupils about how search engines work, point out the ‘sponsored’ results which are shown above or to the side of those generated using this relevance algorithm. The sponsored results are also algorithmically generated, based on the keyword, some quality measure for the advert, the page it points to and often your search history. They are placed on a ‘pay per click’ basis: the search engine doesn’t charge for showing the advert but the advertiser pays when you click on it, so it’s in their interests to only show the most relevant adverts here.

The mechanics will vary from one search engine to another but a good search engine should also: filter out explicit content automatically, allow you to search within a particular site, allow results to be filtered by their location (for example just the UK) and by date range (for example just pages created or edited in the last year).

Classroom activity ideas

- Encourage pupils to use search engines for independent or guided research projects. Get pupils to experiment with the effect that adding in additional keywords or searching for phrases (by putting quotation marks around the phrase) has on a set of results.
- Demonstrate, and ask pupils to use, some of the more advanced search features, such as filtering by date. Show pupils how they can view the cached copy of a web page (for both Google and Bing this is hidden under the green drop-down next to the URL on the results page).
- Read through the Digital Schoolhouse notes on a simulation of how a search engine works, based on Google engineer Doug Aberdeen’s presentation at the 2012 CAS Conference (see Further Resources below). Print off the resources and run this as an activity with your class.

References


