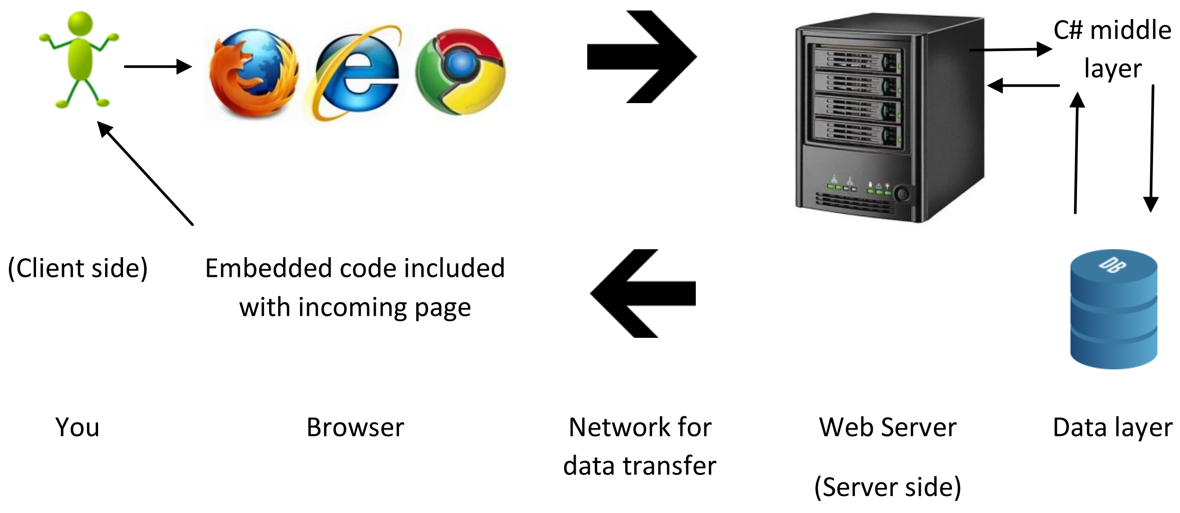
Packet Sniffing

So far we have very much looked at the issues of security from the point of view of one application talking to another without considering the nature of the way data is transmitted over the network.

In the first lecture we looked at a few things that relate to these issues.

We introduced the client server model to see how the client and server communicate with each other…



We have also touched on the idea that the internet is a huge collection of networks talking to each other and in reality it allows one program on one machine to communicate directly with another program on a second machine via a combination of the IP address and the port number.

Ports on a given computer may be specified to allow communications between multiple programs on the same computer e.g.

- 80 HTTP (web pages)
- 21 FTP (File transfers)
- 119 NNTP (Network News Transfer Protocol)
- 443 HTTPS (secure web pages)

Clearly all of the above points hold true and are important but there is another level of detail in TCP/IP that opens up another world of potential issues when it comes to security.

The Network Connections

Assuming you are sitting at home doing something on the internet your local computer will almost certainly make the connection to the internet via your home router. This could be a wireless (wifi) connection or a wired connection.

From here your home network connects to a Wide Area Network (WAN) typically run by your Internet Service Provider (ISP). This may involve a WAN for the area in which you live, which then links to a larger WAN for the whole city. At this point your data is now on the wider internet using high bandwidth routers and high speed connections.

To give you some idea of the number of computers that might be involved in communicating data from source to destination it is possible to use the “trace route” command. The output of this command traces the route of data from your client machine to the server and lists the IP addresses of the machines in between.

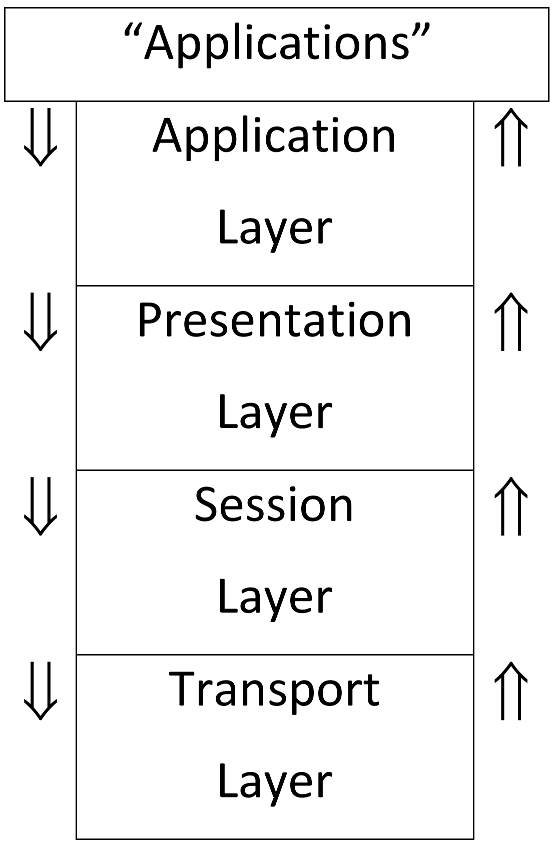
This is all well and good, however the question then arises how does the client machine’s data actually get from the client to the server and back again?

Layered Network Architectures

One way of describing a computer network is to look at it as a series of layers.

One famous layered model is called the ISO OSI seven layer model (you may have come across this already.)

ISO OSI Reference Model



Provides communication services to applications. E.g. file transfers and electronic messaging.

Encoding decoding of character (ASCII EBCDIC) sets & selecting rules to be used. Compression & security (cryptography).

Determines which PC talks to which PC. May manage accounting.

Establishes connections of right type and quality. Initiating data transfer. Managing data to be sent. Releasing the connection.

(The three layers below make up the “sub net” they worry about how data gets from A to B. The layers above are concerned with the meaning of the data rather than its movement.)

⇓	Network Layer	⇑	Concerned with routing data between addresses. Manages the sub-net, e.g. dealing with bottlenecks or formatting issues over different types of network.
⇓	Data Link Layer	⇑	Hides the layers above from the Physical Layer. Concerned with error detection / correction. Sees data as “frames”.
⇓	Physical Layer	⇑	Delivers data from one computer to another. Concerned with placing electrical signals on wires. Sees all data as bits.
	“Cabling”		The physical media data is transmitted upon. E.g. coaxial UTP fibre optic etc.

By splitting the different roles of a network into different layers it is possible to concentrate on different aspects without having to worry about the functioning of other parts.

For example, the physical layer is solely concerned with how the data is represented on the cable as electrical signals. It doesn't care what those signals mean, nor what happens if something goes wrong with the transmission.

The data link layer is concerned with making sure that if something does go wrong there are error correction mechanisms in place to ensure that a noise error prone channel is 100% error free. It doesn't care how the data is being represented on the cable, only that the data gets there safely.

There is a sense in which the 7 layer model is aspirational in that it typically isn't followed 100% of the time. In some implementations the upper most layers may not be implemented at all.

TCP/IP

Another layered network architecture is TCP/IP which sort of maps to the ISO OSI model

OSI Model
Application
Presentation
Session
Transport
Network
Data Link
Physical

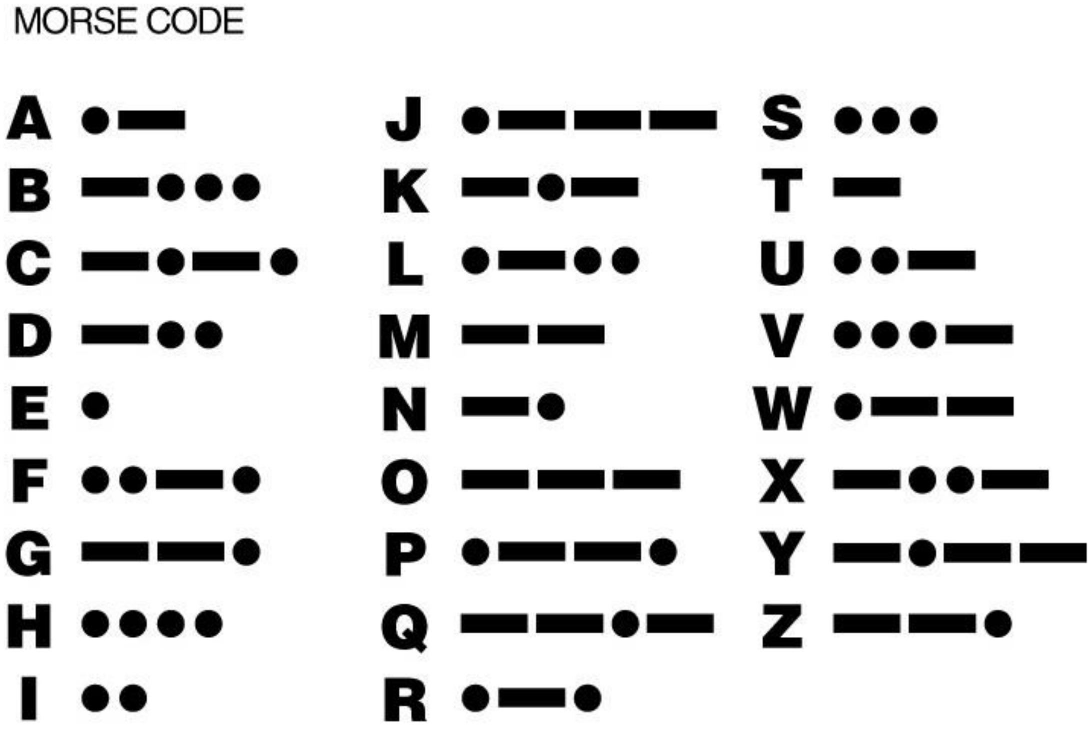
TCP/IP
Application
Transport
Internet
Network Interface
Hardware

Within TCP/IP are a number of protocols that determine how data is transmitted from source to destination.

The Representation of Data on the Cable

The lowest level of data representation on a network is at the level of the "cable" (this may be wired or wireless)

The earliest implementation of this type of system is Morse code



MORSE CODE

A	●▬	J	●▬▬▬	S	●●●
B	▬●●●	K	▬●▬	T	▬
C	▬●▬●	L	●▬●●	U	●●▬
D	▬●●	M	▬▬	V	●●●▬
E	●	N	▬●	W	●▬▬
F	●●▬●	O	▬▬▬	X	▬●●▬
G	▬▬●	P	●▬▬●	Y	▬●▬▬
H	●●●●	Q	▬▬●▬	Z	▬▬●
I	●●	R	●▬●		

Rather than using dashes and dots to represent the data all data is broken down into binary, i.e. a series of zeros and ones.

For example the text “hello” represented in binary using UTF-8 would be “0110100001100101011011000110110001101111”

All data on the computer is encoded using zero and one.

This might then as with Morse code be represented as series of signals, e.g. a beep could be one, no beep could be zero. (It isn’t as simple as that but you get the idea!)

If you have nothing better to do you might check out the following YouTube video of a range of modem sounds.

https://www.youtube.com/watch?v=ckc6XSSh52w

If you hear the sound of an old modem it may bring back some memories but it won’t tell you much about what the data actually means so we need the upper layers to make sense of the noise and turn it into something useful.

This is rather like asking the person who receives the Morse coded message what the message says. The dots and dashes are the raw signal, the person receiving the message and writing down the words would represent an upper layer of the network architecture.

Although from the point of view of the cable the data is a series of zeros and ones from the point of view of the TCP/IP layer the data is viewed as a series of packets. Packets are the zeros and ones organised into a more logical sequence such that they make some sort of sense.

It is a bit like the Morse code operator writing down the message with the letters from the dots and dashes, but the data has yet to be given any further structure e.g. convert the letters into actual words.

TCP/IP Protocols

A protocol is an agreed upon way of doing things. We have protocols in life all the time, e.g. shaking hands with somebody when you meet them.

Breaking a protocol may result in disruption e.g. refusing to shake a person’s hand, or in the case of a computer result in disrupted data transmission.

Internet Protocol (IP)

Within TCP/IP there are a few protocols to consider starting with the Internet Protocol (IP). IP organises the raw data into frames.

IP concerns itself with getting the data to the correct machine. There isn’t a great deal of data included with IP only the destination address and some raw data (that could in fact be anything). Al it cares about is which machine the data is going to.

IP Header	Data

Since the data at this low level is only indicating which machine the data is for, it doesn't identify which application via the port the data is going to. The data could be for a web browser on port 80 or an FTP application running on port 21.

To address this issue we need to impose more structure on the frames by bundling the data together into.

User Datagram Protocol (UDP)

A UDP packet contains additional information for routing the data.

IP Header	Port No & Checksum	Data

This time as well as the destination the packet contains the port number for the destination machine, a checksum (for error checking) and the data we want to send.

UDP is a relatively simply protocol. For example if there is an error in the checksum when the packet arrives then it is simply ignored. UDP also doesn't communicate back to the transmitter that the data has arrived safely.

So what is the benefit of UDP?

Due to its simplicity UDP is fast protocol for transmitting data. It makes little fuss about errors and doesn't bother reporting connection failures, but what is the point?

UDP is an ideal protocol for such things as video conferencing. For example if you have ever used say Skype and the video has gone a bit strange, this is due to UDP packets vanishing or becoming corrupt on route. Do you actually car you friends face looks a bit weird or their voice breaks up occasionally, I suspect not.

Transmission Control Protocol (TCP)

If you are talking to your friend on a Skype call would you be very upset if their voice breaks up occasionally?

What if you were making a banking transaction and half of your bank details were lost en-route? Clearly this would be a bigger problem, so there needs to be some mechanism for creating a communication channel where errors are both reported and corrected.

The packet structure of TCP is similar to that of UDP but it also includes a sequence number for each packet.

IP Header	Port No & Checksum	Sequence no	Data

By giving each packet a sequence number the receiver now knows two things:

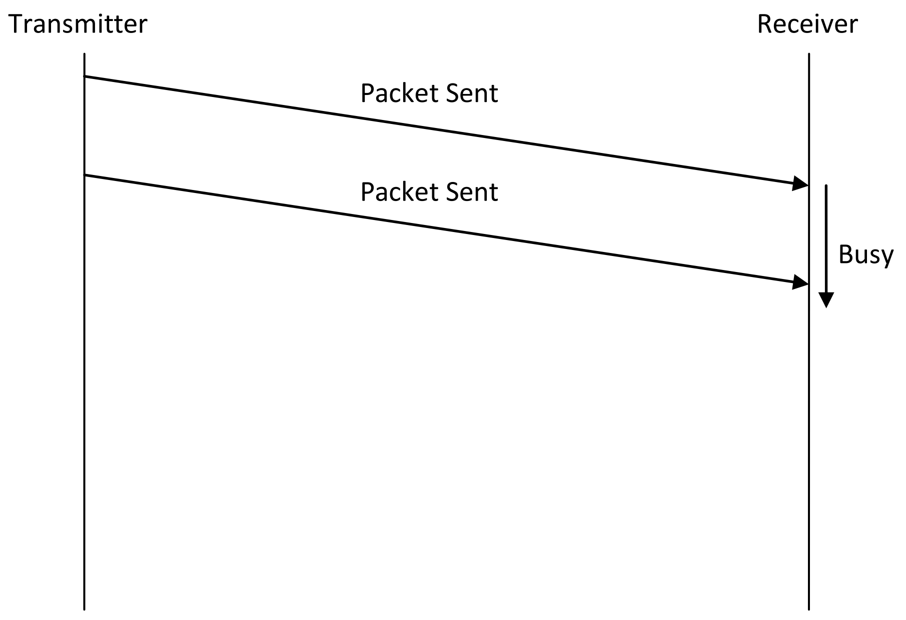
1. What order the packets are supposed to go in
2. If any packets are missing when they are assembled

TCP IP uses a system of error correction called a sliding windows protocol, here are the basics of how this works.

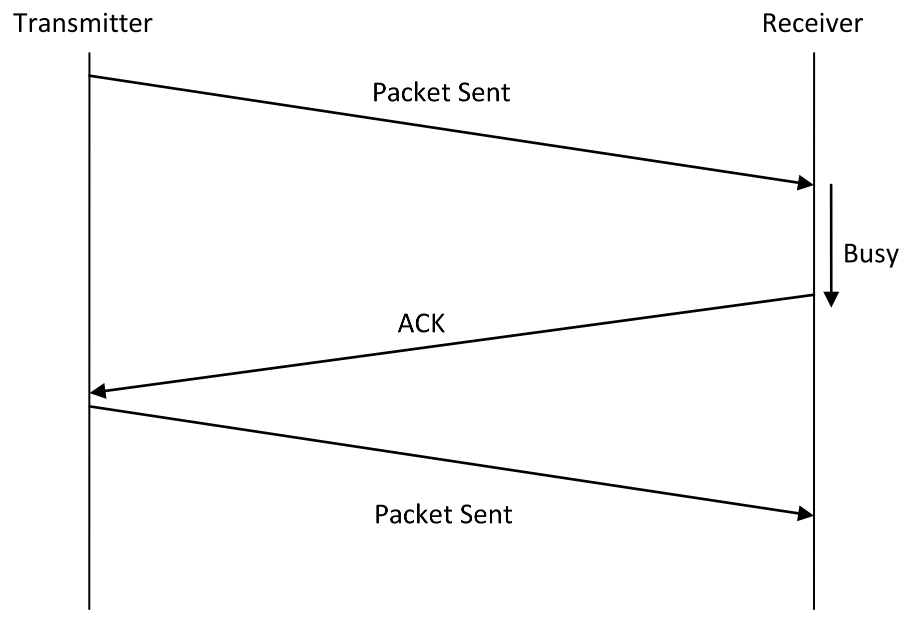
Sliding Windows Protocol

If the transmitting computer sends a packet of data to the receiver there are a couple of questions that might be worth knowing to ensure successful communication.

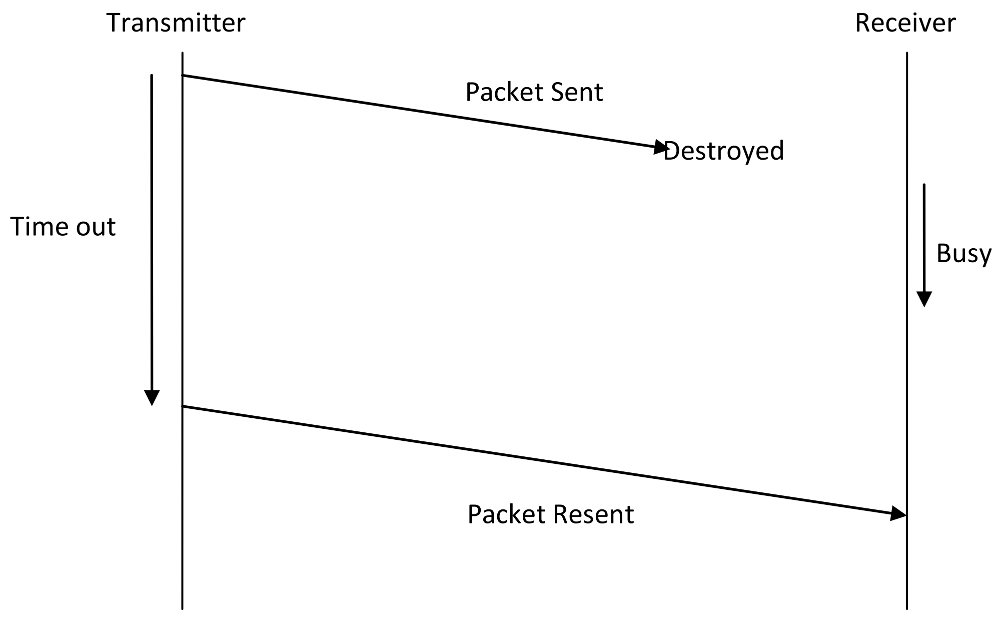
Firstly is the receiver ready for more data, secondly did the data actually get through.



To fix this we arrange for the receiver to send an acknowledgement (ACK) to indicate that it is ready for more data and that the packet arrived safely like so…

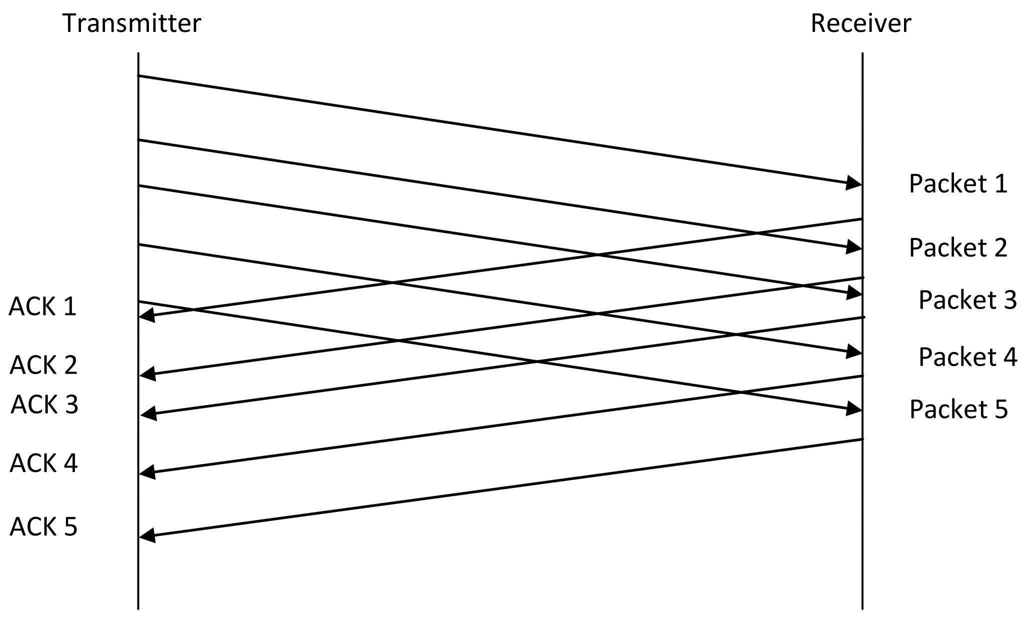


So if the packet never arrives in the first place the ACK never gets sent back and the transmitter knows that the first packet failed to get through…



This gives us some measure of simple error detection and correction. If in this case the ACK is just delayed and a duplicate copy of the first packet is sent it doesn't matter. Since the packets are numbered and the receiver gets a second copy of packet 42 it just ignores it as a duplicate.

TCP Isn't as fussy as this example indicates though. It is capable of having multiple packets in flight waiting for multiple ACKS using a rotating packet numbering system called a sliding window.



This speeds up the whole process by sending more data at the same time.

The Problem with Packets

One of the reasons why we have looked at some of these low level processes is to get a sense of what is going on within the network.

With all of the data travelling back and forth it opens up a potential security hole in the system.

Wireshark

https://www.wireshark.org/

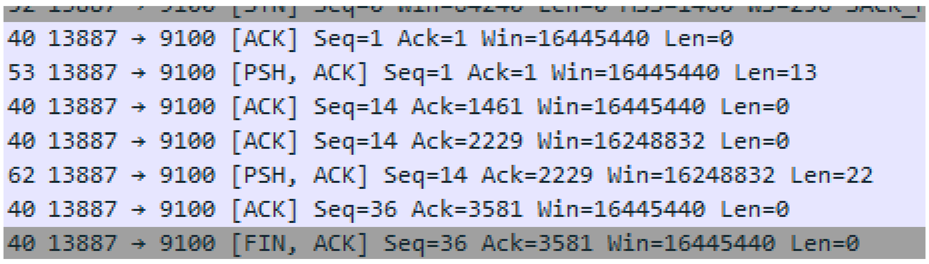
Wireshark is a network packet analyser and may be used to capture the packets travelling over the network typically saving them to a file called a PCAP file for further study.

A tool such as this can be used for perfectly legitimate uses such as troubleshooting network problems, or for understanding network protocols or for “other purposes.”

There are a number of ways you might gain access to the network. You might simply use a machine that is legitimately logged onto the network; you could access the wifi using a hacking tool or you may use a knife to cut a cable and add a new machine to the network which is not authorised.

Below is an example of a PCAP file, having now discussed the nature of the data floating around the network the contents of the data should start to make some sense.

For example we are able to see the ACK packets with their sequence numbers…



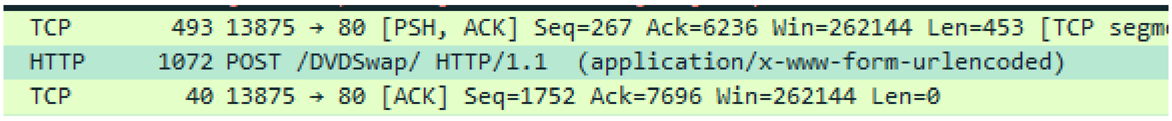
```
40 13887 → 9100 [ACK] Seq=1 Ack=1 Win=16445440 Len=0
53 13887 → 9100 [PSH, ACK] Seq=1 Ack=1 Win=16445440 Len=13
40 13887 → 9100 [ACK] Seq=14 Ack=1461 Win=16445440 Len=0
40 13887 → 9100 [ACK] Seq=14 Ack=2229 Win=16248832 Len=0
62 13887 → 9100 [PSH, ACK] Seq=14 Ack=2229 Win=16248832 Len=22
40 13887 → 9100 [ACK] Seq=36 Ack=3581 Win=16445440 Len=0
40 13887 → 9100 [FIN, ACK] Seq=36 Ack=3581 Win=16445440 Len=0
```

Unfortunately we are also able to see something far more interesting.

When the user of a system enters some data on a web site (say a username and password) the data is also bundled in a packet and sent over the network as plain text.

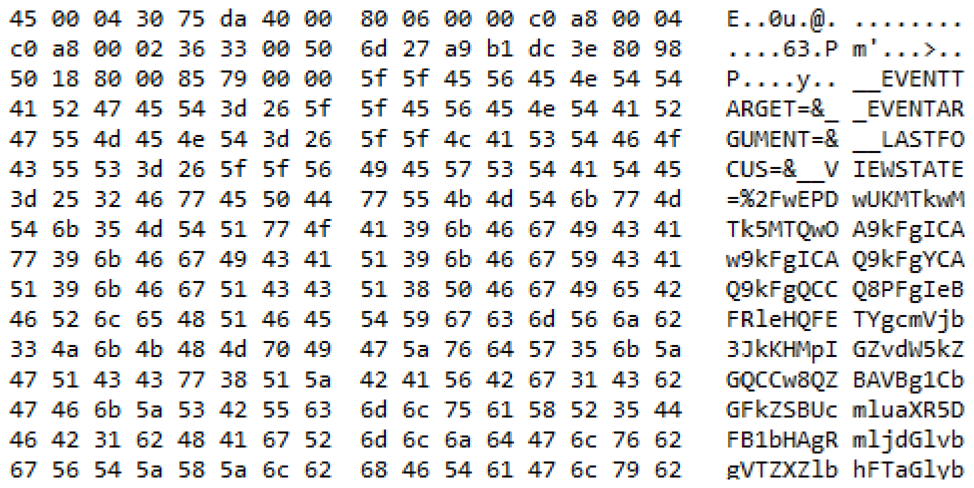
So in the example PCAP file on the web site we had Wireshark running while the user entered their username and password. As soon as they pressed submit a POST request was sent from client to server.

We can see this POST request within the data…



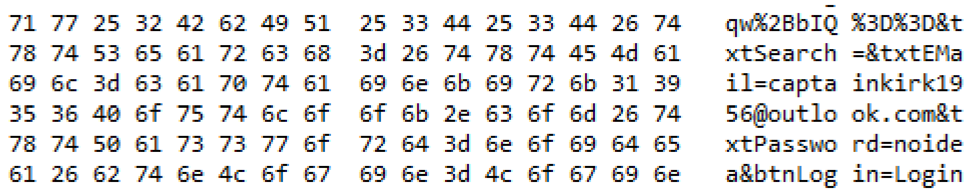
```
TCP       493 13875 → 80 [PSH, ACK] Seq=267 Ack=6236 Win=262144 Len=453 [TCP segm
HTTP     1072 POST /DVDSwap/ HTTP/1.1  (application/x-www-form-urlencoded)
TCP        40 13875 → 80 [ACK] Seq=1752 Ack=7696 Win=262144 Len=0
```

Using Wireshark we may also examine the data stored in the packet which looks like this…



```
45 00 04 30 75 da 40 00  80 06 00 00 c0 a8 00 04   E..0u.@. ........
c0 a8 00 02 36 33 00 50  6d 27 a9 b1 dc 3e 80 98   ....63.P m'...>..
50 18 80 00 85 79 00 00  5f 5f 45 56 45 4e 54 54   P....y.. __EVENTT
41 52 47 45 54 3d 26 5f  5f 45 56 45 4e 54 41 52   ARGET=&_ _EVENTAR
47 55 4d 45 4e 54 3d 26  5f 5f 4c 41 53 54 46 4f   GUMENT=& __LASTFO
43 55 53 3d 26 5f 5f 56  49 45 57 53 54 41 54 45   CUS=&__V IEWSTATE
3d 25 32 46 77 45 50 44  77 55 4b 4d 54 6b 77 4d   =%2FwEPD wUKMTkwM
54 6b 35 4d 54 51 77 4f  41 39 6b 46 67 49 43 41   Tk5MTQwO A9kFgICA
77 39 6b 46 67 49 43 41  51 39 6b 46 67 59 43 41   w9kFgICA Q9kFgYCA
51 39 6b 46 67 51 43 43  51 38 50 46 67 49 65 42   Q9kFgQCC Q8PFgIeB
46 52 6c 65 48 51 46 45  54 59 67 63 6d 56 6a 62   FRleHQFE TYgcmVjb
33 4a 6b 4b 48 4d 70 49  47 5a 76 64 57 35 6b 5a   3JkKHMpI GZvdW5kZ
47 51 43 43 77 38 51 5a  42 41 56 42 67 31 43 62   GQCCw8QZ BAVBg1Cb
47 46 6b 5a 53 42 55 63  6d 6c 75 61 58 52 35 44   GFkZSBUc mluaXR5D
46 42 31 62 48 41 67 52  6d 6c 6a 64 47 6c 76 62   FB1bHAgR mljdGlvb
67 56 54 5a 58 5a 6c 62  68 46 54 61 47 6c 79 62   gVTZXZlb hFTaGlyb
```

Not very exciting, but if we scroll to the end of the HTTP POST request we see the following data…



```
71 77 25 32 42 62 49 51  25 33 44 25 33 44 26 74   qw%2BbIQ %3D%3D&t
78 74 53 65 61 72 63 68  3d 26 74 78 74 45 4d 61   xtSearch =&txtEMa
69 6c 3d 63 61 70 74 61  69 6e 6b 69 72 6b 31 39   il=capta inkirk19
35 36 40 6f 75 74 6c 6f  6f 6b 2e 63 6f 6d 26 74   56@outlo ok.com&t
78 74 50 61 73 73 77 6f  72 64 3d 6e 6f 69 64 65   xtPasswo rd=noide
61 26 62 74 6e 4c 6f 67  69 6e 3d 4c 6f 67 69 6e   a&btnLog in=Login
```

Here we can see the names of the text boxes txtEMail and txtPassword along with the data being sent in the POST i.e. the user's email captainkirk1959@outlook.com and their password "noidea"!

Clearly this is a potential problem!

Secure Socket Layer (SSL)

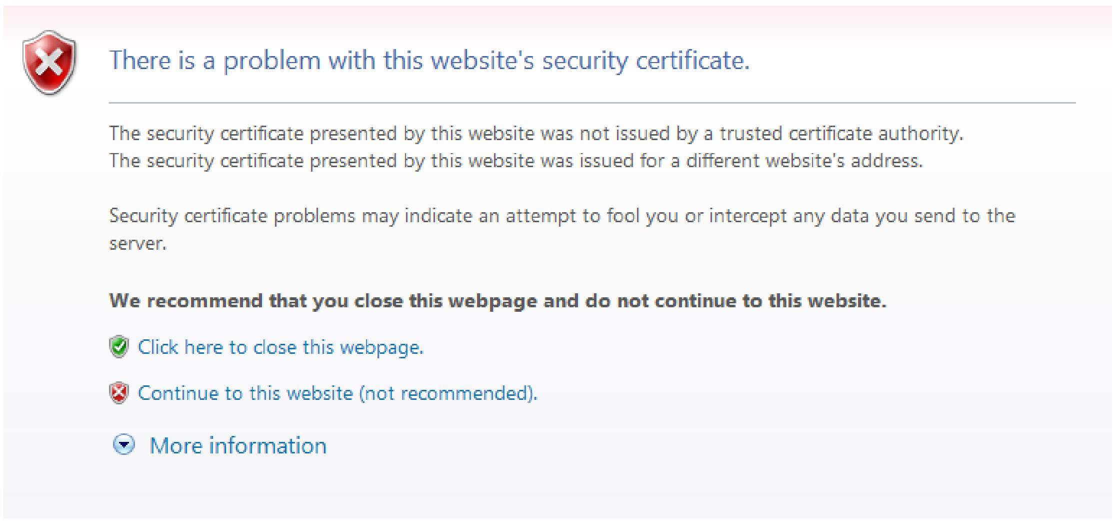
SSL uses a system of public and private keys to encrypt the data.

- The browser makes a secure HTTP request HTTPS on port 443
- The server sends back a digital certificate verifying its credentials
- The client verifies the certificate with the issuing agency
- Using the public key data is encrypted between client and server

The certificate is a file verifying that the server is a trusted source. The certificates are issued by a certificate authority.

Setting up SSL on IIS is fairly simple the downside is that setting up a certificate that is approved by a certificate authority will cost money.

IIS does allow for self signed certificates to be created. These still allow for an SSL connection to be established but may generate the following error in a browser...



There is a problem with this website's security certificate.

The security certificate presented by this website was not issued by a trusted certificate authority.
The security certificate presented by this website was issued for a different website's address.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage and do not continue to this website.

Click here to close this webpage.
Continue to this website (not recommended).
More information

Virtual Private Networks

In a similar manner to SSL a VPN encrypts your data but creates a network within a network.

There are a number of features of a VPN that makes it more secure and private.

- It masks your IP address
- It can make it appear you are in a different country
- Packets may be encrypted

A VPN can be a very useful form of protection if you are using public wifi and your data is being passed around the public network in plain text. All communications in passwords become inaccessible to somebody wanting to examine the packets you are generating.