Right of privacy reconciled with right to information

The purpose of this article is to describe, in the context of current public expectations about technology such as phones and computers, a right of privacy which appears to be consistent with a right of access to information.

I'll begin with a number of anecdotes; these are not carefully researched, and the intention is not to criticize or condemn any of the particular companies that are mentioned.

1. Airbus.

For the first two anecdotes, I'll mention two Airbus crashes. The first, when the aircraft was first being demonstrated by a test pilot, occurred when the pilot had dived towards a woodland area. When he pulled up on the control stick, expecting the aelerons to move to an intended upward angle, his actions were interpreted by a computer algorithm. The algorithm included aircraft speed and other variables, and within the algorithm, unknown to the test pilot, was a maximum allowable g-value. There was no configuration of the controls which would move the aelorons to the intended angle; they were temporarily constrained to a small range of motion at that time, and the aircraft crashed into the woods dosing the woodland with jet fuel, which then burned.

2. Airbus again

For the second anecdote, during a later flight on an airbus, the pilot allowed his young son to play with the controls. It was not true, in this later instance, that no motion of the controls could affect the aelerons and rudder. In fact, the situation was somewhat the opposite. The aicraft was in autopilot, and the pilot believed that the controls were completely disabled. However, now the algorithm included a sensor to detect forces on the controls hard enough to imply an attempt to over-ride such limitations as had caused the first crash. When the child played forcefully with the controls, the aircraft came out of autopilot, responded to the forceful actions with accelerations of high g forces which caused the aircraf to spin and dive uncontrollably, causing a second fatal airbus crash.

3. Audi.

The third anecdote is merely personal and involves an Audi car. My wife noticed that an indicator light known as the 'glow plug warning light' had begun flashing in her car. The information of what fault condition has been detected is available by a protocol known as 'can-bus.' This can be read by various devices, such as a cable with a programmed ROM (read only memory) chip, which can for example convert the protocol into the protocol of a COM port or a USB – these are the two prevalent protocols of a personal computer. I purchased such a cable from Amazon, it came with a computer program to convert the data to plain text. The instructions said that the date of the computer must be set to a date in the past (October 2010), and to disconnect any internet connection of the computer being used; and it soon became evident that a component of the cable program had been written by Mr. Ross, who also sells his cables for a price which is comparable to the price of an entire used car.

I mistakenly believed that the windows firewall would block incoming connections; and although I checked that the installation .exe file had not created any exceptions, I neglected to turn on the firewall. I eventually found that Mr. Ross' website had instructed the program to use the USB to reprogram the cable, and delete a code numer which should have indicated that the funds had been paid to Mr. Ross. Subsequently, when the cable was attached to the USB port, the component of the program written by Mr. Ross caused the program to exit with an error message related to the failure to make the payment to Mr. Ross.

A few days later, while my wife was driving her Audi car on a motorway, with a child in the car, the algorithm in the car's controller imposed a limitation which disabled the car from exceeding a low speed (about 35 miles per hour). I learned that this is a feature of the car's controller known as 'limp mode' which activates when the flashing glow warning light has been ignored. There were no cars approaching from behind; if there had been, the car's unpreventable deceleration could have caused a crash, as the limp mode speed of the car is lower than a safe driving speed on the motorway, and no hard shoulder or 'breakdown lane' was available.

A telephone call center advised me that the Audi dealer would be willing to connect a can-bus cable of his own to the car and tell us the meaning of the flashing glow warning light. However, in truth, the dealer asked us to wait in his dealership nearly a whole day. He told us that as a free service, my wife's car will be washed and also vacuumed. We were given coffee and treats, and allowed to sit in a plush waiting area, where new cars were lined up, and there was nothing to look at for several hours besides the new cars.

Eventually we were brought the 'report' from the can-bus cable. The report was an invoice for the cost of repairing the car, and listed a number of parts that would be replaced, the number of hours of labour that would be done, and itemized the cost of the parts and labor. It did not inform us what had been the error which the can-bus cable had indicated, the existence of which was indicated by the flashing glow plug warning light.

It is my belief that the experience of the car's failure, of the day being given treats and coffee, and pampered, and finally given the bad news about our existing car, were essentially an unethical marketing tactic, which endangered the life of my wife and the child whom she was carrying in the car. Just as in the ase of the airbus failure, the danger can be traced to a failure, in this case, my failure to turn on the Windows firewall which would have prevented Mr. Ross' website from instructing his program to deprogram the cable.

4. Tesco.

For a fourth example, one from today actually, I'll discuss the Tesco Hudl. This is an android device which Tesco offers at a low price, it is advantageous to Tesco because it contains branding which is difficult to remove.

My child wished to play a verson of Angry Birds on his Hudl. I had found the .apk file written by the author, and I knew that Tesco had declined to include a file manager on the Hudl.

I put the Angry Birds .apk file on the Hudl, and when my child clicked the 'downloads' button, it showed that the downloads folder is empty. Then I put the es file browser .apk file in the downloads folder. Of course, again, the downloads folder was empty. The on-board file browsing capability of the Hudl recognizes only a limited number of 'file extensions' and, similar to an earlier Microsoft idea, the Hudl itself decides what action to take based on the file extension.

Since it is not possible to see the contents of the Downloads folder without using a file browser, while the file browser's .apk file was there in the Downloads folder, it was not possible to break the loop of inability. There was no way to play Angry Birds.

The solution would be that the Hudl allows a user to go online to the Google Play Store. There, if we had been in a village which has internet service, we could have had the programs in the Hudl 'install' the file browser from Google play. Thereafter, we would have been free of Google play. But, initially, it is not possible to use the Hudl as a universal computing device without first having permission from Google. This is despite the fact that the Android system of a Hudl is completely open source and java-based.

5. Microsoft.

For the fifth example, this is an old one, and more basic than the Tesco example. It concerns the historical relation between the computer and the internet.

The first computers were Turing machines, and the first large computers were loaded using stacks of punched cards. The stacks were kept together, in boxes known as 'files.' These are exactly like the filing cabinets that are still used in offices, and the concept of a digital 'file' was invented by IBM (international business machines corporation).

A 'file' was not only a stack of cards. Rather, what had been the paper tape of a Turing machine (and later became also the magnetic tape in an IBM computer, and in the DEC computers actually was a punched paper tape) represented a sequence of integers; and a 'file' was a subsequence, usually, but not necessarily, a consecutive subsequence.

Thus, the function of a turing machine could be modulated not only by completely replacing the paper tape, but by effectively replacing just parts of the tape.

The UNIX system in the IBM computers, which was later copied and became the MS DOS system, the original prototype of what became both 'windows' and 'mac' systems, labelled files by a name consisting of two words separated by a period.

The idea of the internet is, in the simplest form, that information can be stored in a shared protocol, where it is guaranteed that what is seen by one person, using one computer, is identical to what is seen by another user. The source of the protocol would be plain text which any concerned user could examine, and the browser which converts the plain text to images or other visible things, was a program resident on the computer which the user could examine, delete, or change.

The Microsoft idea, in 'windows 95' was that the owner of a computer should no longer know the part of the filename after the dot. This was to be secret – although a sufficiently capable and outraged user could change settings in the 'control panel' of the operating system to remove this level of secrecy. Secondly, that the resident program (browser) would not be removable by the user, it would be called 'internet explorer' and would be part of what was called 'windows explorer.'

Thirdly, that except for a very intelligent and determined user, it should be impossible to create any text file which has the name ending in '.html,' the agreed name of the shared internet protocol.

The ways that Microsoft made it difficult to create a text file ending in .html are visible even now in the computer which I'm currently using, twenty years later. If I use the Windows text editor called Wordpad to write a file, and attempt to 'save' the file (to write it to the storage device), I use the mouse to select 'save as.' A box appears saying 'save as type' which includes four types:

Rich Text Document
Text Document
Text Document MS Dos format
Unicode Text Document

If I name the file 'file.html' and choose 'Text Document' it changes the filename to 'file.txt.' Then when I save it it saves the file with the name 'file.txt.'

For a user who has not gone to the control panel and un-checked 'hide extensions for known file types' the change of filename is hidden from him, and irreversible.

But, I did know to un-check 'hide extensions for known file types.' and I can see when I look at the icon of the file that it is named 'file.txt.'

When I click OK, a warning appears, saying

> You are about to save the document in a Text-only format, which will destroy all formatting. Are you sure you want to do this?

Actually, it will not remove the HTML formatting in my document to click 'yes,' but I have to know that Microsoft has lied.

So far, I have needed to make a change in the control panel, I have needed to choose 'text only,' and I have needed to disregard a lying warning, but I am not done. The file which I have saved has an icon now (on the 'desktop') which, because I know to have un-checked 'hide file extension,' I can see is 'file.txt' instead of 'file.html.'

If I know about renaming files, I can try to rename it. Right clicking the icon with the mouse gives options including rename.

The part of the filname that is highlighted does not include the extension '.txt.' But I know how to fiddle witht the mouse to highlight only this part. Many users would not know this.

Changing '.txt' to '.html' causes yet another error. It says

> If you change a file name extension, the file may become unusable. Are you sure you want to change it?

Choosing 'yes' finally finishes the process of writing the ordinary text file to the hard disk in a way that the name will be recognized by a web browser.

Yet, again, unless one has known to do more, the icon will now be a blue e, and if the icon is double-clicked, it will 'open' with internet explorer. If the file includes recent additions to HTML such as javascript, there will be further error messages and warnings, and the internet explorer (which is not a correct browser) is in danger of revealing contents of the file to others, or indeed saving files of others to the storage device without my knowledge or permission, executing programs etc.

Thus, the fifth example of a violation of open source protocol is Microsoft's attempt to obfuscate the distinction between the internet and private files, leaving the final action of any file (on the internet or on a private computer) in the hands of Microsoft themselves.

6. Sony.

The sixth example was my attempt recently to see what is on the storage device of an Xperia phone. I did not realize that simply connecting the phone to a computer by the USB cable when the phone is turned on would allow the USB to identify the device. Rather, the instructions indicated that it is necessary to instal the 'driver' which is a huge suite of software known as 'Sony PC Companion.' The installation instructions ask you to ignore Microsoft's pleading alert to allow Microsoft to install a driver instead. The phone does not turn on without a SIM card, and with a vodafone SIM card, all three providers (Vodafone, Microsoft, Sony) are simultaneously generating conflicting and lying alert message. The device is after all an android device, and like the Hudl it included no file browser of its own.

Discussion.

What all these examples have in common is that the user has in his mind an expectation of really two protocols. The Turing machine protocol, where a program on his device consists of a sequence of instructions, and subsequences filling up the sequence (with none missing) are given filenames which are completely known to the user. And the internet protocol, where a user can generate information which he wishes to be shared, and place it publicly on servers.

In all six examples, what takes place is that the user is deceived about the nature of the protocol. In the case of the example of the Hudl, the 'downloads' button uses the scheme invented by Microsoft of hiding from the user some of the subsequences of his turing machine sequence. Here, not only is the extension '.apk' hidden, in fact any file which has the extension '.apk' is not shown as belonging to the 'downloads' folder when the user clicks the 'downloads' button.

The user's expectation is that his storage device is a disjoint union of 'files', that these files have filenames, and that those names are partitioned into 'folders' or 'directories.' This can be made to be true, using an interaction with Google Play, and in fact the user manual recommends this action to take place. But, without interacting with Google Play, the device is not a Turing machine with all its data knowable by the user.

In the case of the Airbus example, the first pilot who died assumed that the algorithm on the device would allow him to control the aeleron angle. The second pilot, who died along with his child, copilot and passengers, assumed that when the plane is in autopilot, the algorithm would not.

The Audi example is more complicated, the way in which my failure to turn on the firewall ended up with the car unpredictably and unpreventably going into limp mode on a motorway. The proprietary Volkswagen-Audi protocols were made public by Mr. Ross, and he had lost ownership of the protocols. He was able to essentially get revenge upon us by instructing his code to overwrite the programming of the ROM chip in the cable which I had bought.

In many of the examples above, a public right has not been completely removed except in trivial ways. Someone using a Hudl needs only an inessential contract with Google, currently, before the device will function as a turing machine.

However, such trivial examples if they are allowed to persist in the hardware level would remove the right of a user to expect that his device (telephone, home computer, etc) could be presumed to be a turing machine with all parts known. Subsequent violations of privacy, such as already take place with computer viruses, would then be unpreventable, as the user would not be able to distinguish his private information from his public information. Thus if DRM ever enters the chip level – to the point where protocols can be hidden more deeply in the microscopic structure of the hardware chips – an underlying and unremovable deception will become unpreventable.