**The Future of Security**
**Professor Richard Harvey FBCS**

25 May 2021

I titled this lecture the Future of Computer Security and I'm now worrying that the title was glib: no-one knows the future of computer security, because there are variety of unpredictable factors that will affect the technology we use to protect our devices. That said, computer security has been and is likely to remain to be, a function of four parameters:

1. What a particular society deems as private;
2. The nature of the goodies on offer to criminal hackers;
3. The technical difficulty of the hack;
4. The connectivity of the machine or user under attack.

The fourth factor is routinely ignored since the presumption in modern society is that every machine should be fully connected to the internet[1]. As we saw in the previous lecture, the basic transmission mechanism in the internet, TCP/IP, is not only unencrypted but also sends transmission and reception addresses in the clear. This incredible candour is one of the internet's major downsides but its convenience, power and speed trump everything else. So, for the rest of this lecture, I shall assume that the obvious solution, which is to disconnect, is not viable.

The first factor, the nature of privacy, is a deep question and one that is highly variable by society. Gabriel García Márquez is famous for his observation that "Everyone has three lives: a public life, a private life and a secret life." Much journalistic ink has been spilled over what constitutes the boundaries of those lives. Both the Royal Society (Protecting privacy in practice) and the Royal Academy of Engineering (Dilemmas of Privacy and Surveillance, 2007) have studied that matter and reached consensus that Privacy is a good idea, and we ought to develop technology that helps people preserve it. And Gresham College also has a back catalogue of lectures on the topic. As far as the law goes then even Article 8 of the European Convention on Human Rights ("everyone has a right to a private life") is much disputed. Most countries interpret the Article mean "You have right to a private life except when we say so," and in Kafkaesque twist "we will not tell you the circumstances in which we may override you right to privacy in case it restricts our freedom to spy on you." I have nothing clever to offer on this topic other than there are too many people debating this topic who do not seem to understand the basics of computer security.

The lecture is about the second and third factors: there is a trade-off between the rewards for computer crime and the effort to which criminals are prepared to go. This trade-off is well known in security circles and many cryptographic systems are designed and updated using estimates of computing power that might be available to criminals. In my view, the rewards have changed rather dramatically in the last few years and in this lecture, we examine the scales of technical complexity versus reward and see how the balance might tip.

[1] There are a few exceptions: a few organisations run entirely separate internal wired networks but even then, it is less than rare to find someone has temporarily disconnected the secure machine from the secure network and connected it to the insecure one. Such stories are commonplace – a former cabinet minister couldn't get his PC to work with the cabinet office network so bought an Apple Airport and plugged it into his office ethernet port. The CEO of a very large networking company was annoyed to find his new iPhone was blocked by his own SYS Admins due to potential insecurity in the early iPhone. The SYS Admin rather reluctantly gave him a work-around and swore him to secrecy. Within seven days there were 10,000 iPhones registered on the network!

On the subject of technical complexity, the root cause of most security problems is that developers make errors. There are some excellent Gresham lectures on this topic by the former Professor of IT, Martyn Thomas. He discusses what errors developers make, how often they make them and what can be done about it. The latter point is where there has been considerable progress over the years. The major consumer computer companies: Amazon, Alphabet, Apple and Microsoft all have extensive code review and bug-finding activities.

Bug bounties are an exciting new idea – companies offer very substantial rewards ($1M or more for serious bugs) to developers who spot security flaws in companies' products. Bug bounties offer a twin advantage of mobilising large numbers of people, without the strictures of corporate group think, to look for flaws but also it provides rewards for "white hat" hackers and thus prevents them becoming "black hats".

The theme of reward versus effort has an ancient history in cryptography. The basic philosophy is that no code is uncrackable – it merely has to resist attack for just long enough. Cryptography is one of those fascinating topics that is very theoretically complex but practically very simple – if two people Bob and Alice[2] share a number that is known only to them, a secret key, then they can agree to use any number of algorithms which munge their messages into something uncrackable – if the interloper, Eve, knows the key, and the algorithm, then it is trivial for her to decode the message. If she knows just the algorithm, but not the key, then it's impossible. And every time I open my banking app, or make a web transaction, that is what happens. And as Edward Snowden said "Encryption works. Properly implemented strong crypto systems are of the few things you can rely on…"

But how did Bob and Alice, get that shared secret? The answer is public key cryptography. Alice and Bob choose a private key (often using a random number generator). They then use an algorithm known as the Diffie Hellman Merkle key exchange algorithm. It works like this:

Firstly, all these algorithms have parameters. In the case of Diffie Hellman Merkle there are two parameters *G* and *p*. *p* is usually some monstrously large prime number such as

155251809230070893513091813125848175563133404943451431320235119490296623994910
210725866945387659164244291000768028886422915080371891804634263272761303128298
374438082089019628850917069131659317536746955176311984337163722100721057791 9[3]

And *G* can be quite small and is usually a primitive root of *p*. For this *p*, *G* = 2.

If Alice's private key is *a*, then she computes a public key as $A = G^a \bmod p$. Without even understanding the mathematics we can see this looks like a pretty simple equation. $G^a$ means we multiply *G* by *G* by *G* …we do that *at* times. And mod *p* means we divide the result by p and return the remainder. That modulo part makes it very tricky for Eve to work backwards to *a*. To think about a little example in the decimal system, consider that 17 mod 11 = 6 (17 can be divided by 11 once leaving a remainder of 6). Thus 17 mod 11 =6 but also 6 mod 11 = 6, 116 mod 11 = 6, 1358024685 mod 11 = 6 and so on. There is an infinity of possibilities for *a*. Even if *a* is known to lie in a range, can you spot a pattern in the numbers 17, 116, 1358024685? Solving for *a* in general has, so far, no efficient algorithm.

Bob sends his public key $B = G^b \bmod p$ which is the same equation but applied to *b*. Alice now computes $s_a = B^a \bmod p$ and Bob computes $s_b = A^b \bmod p$. With a bit of mathematical substitution, you should be able to show that $s_a = s_b$. Both Alice and Bob have a secret number. The number

[2] The world of cryptography has its own list of characters. Bob and Alice are always innocents trying to communicate. The eavesdropper is usually called Eve or Yves.

[3] Actually, don't use that number, it has probably already been hacked. Pick a bigger one.

depends on their private keys *a* and *b* (and *G* and *p*) but Eve does not have enough information to compute *a*, *b* or $s = s_a = s_b$. What a beautiful algorithm – who would have thought that one simple little equation could be so difficult to reverse engineer.

Should Eve attempt to reverse engineer Diffie Hellman Merkle then the conventional wisdom would be that she would have to solve the discrete log problem. The fastest algorithm known is the number field sieve algorithm which, even when implemented very carefully, is slow. But how slow? In 2001 two authors, Arjen Lenstra and Eric Verheul decided to make some estimates. They assumed that computing power would double every 18 months (Moore's law). For a system in use in 2021 with a life of 1 year (ie 2022) they predicted Diffie Hellman Merkle would need a size of around 1995 bits[4]. In practice this is getting a bit unwieldy so, in the lecture I show how the Diffie Hellman Merkle protocol can be modified to use elliptic curves. These are more difficult to crack so we can use less complicated parameters.

The Diffie Hellman Merkle protocol is widely used so we should hope that a team of scientists were wrong when they claimed in 2015 to be able to crack 512bit keys in a few hours (Adrian, et al., 2015)[5] but it is vulnerable to something called a "man in the middle attack". When Alice tries to communicate with Bob, unknown to her, her internet packets go via Yves. Yves pretends he is Bob and Alice establishes secure communication with Alice. Likewise with Bob. Yves can now pass traffic from Alice to Bob and vice versa – the traffic looks genuine to both parties, but Yves sees everything in the clear. Truly on the internet nobody knows you are a dog (or an imposter). The solution is to issue Bob and Alice with digital certificates which contain their public keys. When Bob supplies his public key to Alice, Alice checks via a Certification Authority that Bob's key has been issued to Bob. Even this process has some subtlety. In practice Bob supplies a certificate that is "signed" by a Certification Authority. I am looking at the Gresham College Website now and I can see that the certificate associated with Gresham is signed by Amazon, it was last renewed a couple of hours previously and when I decrypt it, using the Certification Authority's public key, I can check that its *hash* matches which implies that the information in the certificate has not been tampered.

Hash algorithms pop up all over the place in computer science. Let's imagine I am transferring a large file to you. How do know that your version of the file is the same as mine? We could go through bit by bit, but that would take ages and might introduce errors – what if we make an error in our checking? Maybe I could add one extra bit on the end of blocks of the file, or the whole file, such that the total number of ones was even? That is what is called even-parity. If one bit is flipped then the parity will be wrong your end. Well maybe I add up the total number of ones: a checksum? TCP/IP the internet protocol uses checksums in each packet – if the checksum does not match then we are asked to resend the packet. Alternatively, I could generate a code, of fixed size, that is a fingerprint of the data. If the data changes, the fingerprint changes. That fingerprint is called a hash. A good hash algorithm will be quick to compute; will look random (even if the input data are not), will be of fixed size; will be sensitive to change (if one-bit changes in the input data then the hash should change a lot) and it will minimise "collisions" (two sets of input data with identical hash values). In the internet, where everything is readable by everyone, we must also demand that the hash cannot be reverse engineered to give the data – this is the so called cryptographic hash. Certificates are signed with cryptographic hashes – in the case of Gresham College certificate, Amazon signed it with a well-known public algorithm called SHA-256.

All of which is classic computer security, by which I mean, were we teaching a computer security course 20 years ago then the same principles would have applied. We were using shorter keys back then, because computers were slower and we were using simpler encryption, because computers were slower, but essentially the enemy consisted of people with time on their hands and some

[4] There have been several recommendations since then – they are all summarised at an online calculator at www.keylength.com

[5] At least two authors have claimed they are wrong.

computer expertise: teenagers, nerds or, most awesome, nerdy teenagers. But recently we have had two major changes: the Snowden leaks and Bitcoin.

I am in no position to know if the Snowden leaks were true, but certainly the US government has reacted as if they were true. Snowden's description of the security agencies as code breakers is not new at all – that is what we expect them to do[6]. His revelations certainly surprised people and a number of experts have read between the lines to conclude that some well-known prime number groups used for Diffie Hellman Merkle are no longer secure. However, he also posited two new roles for the security services. The first of those one might describe as industrial espionage – buying or persuading manufacturers to add backdoors to their systems so that government agencies can access information that the general public thought was secure. This has led to a special mistrust of any algorithms described, designed or mandated by the US government. The second activity is government sponsored hackers looking for vulnerabilities. This raises a fascinating dilemma. When a government agency discovers a vulnerability should they keep it to themselves hoping that it will be useful when they come across a baddie or do they disclose it to the manufacturer because their own government is at risk if it is not fixed. In the US this has been formalised in the Vulnerabilities Equities Process[7]. It is helpful that there is now a formal process but there is a tacit understanding that there are bugs in software we are all using which are known to the security agencies but not the manufacturers. The general assumption of computer security is that, if something is known to the goodies then we should assume it is known to the baddies. An unsettling thought.

The second factor which has to potential to dramatically change computer security is money. Previous Gresham lectures have covered ransomware attacks in which a user's systems are corrupted or impeded until money is paid to the ransomers. The most recent of these is the Colonial Pipeline hack which took place on 7th May 2021. The hackers, known as Darkside, are reputed to extorted have $5M[8] to restore order to the pipeline which supplied petroleum from Texas to the New York region. Both the US and UK governments state in rather anodyne terms that paying ransoms is a bad idea but neither seem to have made it illegal, so the practice continues, and ransomware continues to be a problem.

But electronic money brings new cyber-risks. In many countries those risks are covered by the Bank[9] so it is reasonable for customers to think that their bank is sufficiently incentivised to prevent fraud. But the new cyber currencies, such as Bitcoin, Namecoin, and many, many others have no bank. If bitcoin is hacked, then everyone loses all their money. This is a pretty frequent occurrence and website BlockChain Graveyard is a depressing catalogue of catastrophic failures of blockchains. But what is a blockchain?

Bitcoin and other currencies secure the cash through a ledger. The ledger keeps a record of every payment. To prevent the ledger being corrupted there are multiple copies (hence the term "distributed ledger"). The ledger is split into blocks and for a block to be valid it must contain a special number such that when one computes a SHA256 hash of the whole block it contains a fixed number of zeros in the first, say, 30 positions. You might complain that computing that special number is fiendishly difficult and it is. Bitcoin miners compete to compute those numbers and are paid if they are the first to do so[10]. Blocks also contain the hashes of previous blocks so any attempt to alter a block means altering all the blocks in the chain – a horrendously expensive task. Of course, people might just attempt to change the latest block. Imagine Bob decides to fool Alice by sending her a

[6] Hackers on the side of the goodies are called "White hat" hackers and those on the side of the baddies are called "Black hats" after a supposed analogy to cowboy films where the villains and heroes were designated by the colour of their hats. Whether they are white hats or blacks hats the revelation that governments employ software engineers and cryptographers to break codes is "old hat".

[7] The UK has a similar process that is described on the GCHQ website.

[8] This seems quite cheap to me – at least one other source is quoting $90M.

[9] In the UK the first £75k of cash in the bank is covered by a guarantee.

[10] They are paid in bitcoin though which one might regard as a mixed blessing.

fraudulent block. She adds that to her blockchain but soon enough in come conflicting blocks from other miners. Alice forks her block chain and waits. Now Bob is a race against all the other miners in the world -- he has to keep computing all the subsequent blocks long enough for Alice to keep the fraudulent chain alive. It's a lot of computational work and the proponents of blockchain currencies fervently hope that the work is too much for the reward[11]. The distributed ledger is an intellectually appealing idea, and it has been picked up for many applications. However, as with cryptography, the insecurities mostly arise not from the blockchain but from how it is used.

To access your bitcoin, you need an address in the ledger and its associated private key. The process is reasonably secure on paper, you push your secret key through an elliptic curve crypto system to generate a public key, that is then transformed to a fixed length via a hashing function. So far, so good. The problem arises when I write down what a typical private key might look like:

E9873D79C6D87DC0FB6A5778633389_SAMPLE_PRIVATE_KEY_DO_NOT_IMPORT_F4453213303DA61F20BD67FC233AA33262

What a monster! How on earth are people meant to remember that? So, most people use cryptocurrency *wallets*. The idea should be that your private key is strongly encrypted and can be unlocked with a passphrase but there are good and bad wallets. One of the bad wallets, Brainwallet, encouraged you to think of a plain text pass phrase such as "correct horse battery stable" it then uses the secure hashing algorithm SHA256 to create a key. SHA256 is, or was, a well-respected secure hash so it would be fiendishly difficult to invert it, so no problem, right? Wrong! SHA256 is a very fast hash, and, in this case, we do not need to invert it, we are interested in the forward hash. So, let's write a program to systematically search through plaintext pass phrases, run them through SHA256, encrypt them and see if they match any addresses. Even more damaging, let's precompute as many addresses as we can – say several 100 billion and when that address pops up (Bitcoin changes the addresses frequently), we will be able to raid the wallet because we have the address and its associated private key.

In the lecture, I reference a talk by Ryan Castellucci and Filippo Valsorda at the 11th Hope Conference in 2016 where they explain exactly how this works and provide multiple other bitcoin hacks via their open-source software, Brainflayer (Castellucci, n.d.). Of course, computation is an issue, and the authors needed to spend $50 of their own money on Amazon's compute cloud to do the searching but this talk exposes a very important concept that I believe is under-rated by the blockchain advocates. If you decide to hack my bitcoin, then you have to discover my private key. Even if I was silly enough to use Brainwallet, it is still a horrendous task to guess my passphrase. But distributed ledgers publish all the public keys as soon as they are used. Trying to compute passphrase that match *any* of the available keys while not trivial is so, so, much easier. And it's worth the effort because every bitcoin address contains money. In mathematics there is a delightfully simple analogy known as the birthday problem. A teacher with 23 children in the class wonders what the probability is that one of the children shares his birthday. The answer is an unsurprising 0.06. But what is the probability that any two pupils share a birthday – 0.5 is the answer. It is so much easier to find matches among two lists than it is to match a specific item. Public ledgers make available lists to attack and the attacks on bitcoin have been legion – reddit is full of people complaining that they have been robbed and often of tens of thousands of dollars.

This is not the only hack of bitcoin – the most famous attack was that perpetrated on Mt Gox which was the dominant bitcoin exchange in 2014. The jury is still out on what happened at Mt Gox – certainly several hundred million dollars was stolen partly due to the discovery of the Mt Gox private

[11] There is an excellent explanation of BlockChain by Grant Sanderson on the 3Blue1Brown YouTube Channel at https://www.youtube.com/watch?v=bBC-nXj3Ng4

key via hacking and partly because earlier versions of bitcoin did not compute the block hash properly[12]

I have talked about the two big changes to the context in which computer security must function. The first were state actors – intelligence agencies and so on. While there is some hope that the Vulnerabilities Equities Process might regularise what is a highly unregulated activity, there are still plenty of naïve law enforcement officials who see end-to-end encryption[13] as a threat. A typical argument would be that end-to-end encryption is a threat to the detection of child exploitation/modern slavery/terrorism (insert moral panic here) and that it must be stopped. What they ignore is that the internet is an unregulated packet-based system where anyone can see anyone else's packets. Replacing end-to-end encryption with backdoor encryption or weak encryption is a dream for financial criminals. It is becoming a tiresome routine to see an Interior Minister dragooned into appearing before the Press to moan about end-to-end encryption. It is the equivalent of moaning about door locks because drug dealers use them to protect their properties from police raids.

The second big change is the use of attacks that are triggered by state actors. One might hope that state security services would be highly targeted in their activities. Doubtless some are WannaCry, as described by Gresham Professor Tarah Wheeler (Wheeler, 2021) were indiscriminate. The WannaCry hack attacked everyone – most people were protected but of those who are not, some had money and presumably paid up. Wheeler made a very good case that such attacks are equivalent to war crimes. More recently we have seen the SolarWinds Sunburst hack cause considerable alarm. Solarwinds is a company that sells software tools to help large organisations manage their networks. Like many such companies it distributes periodic updates to its customers which are signed with a digital certificate. It seems a group of hackers, probably state sponsored, breached the Solarwinds ftp server and modified the updates so that they could have superuser access to the machines and hence networks. Since the Solarwinds machines were compromised it was possible to fool customer machines that the updates were genuine since they were signed with genuine certificates. Unfortunately, those updates contained malware and tens of thousands of machines downloaded malware. Such "supply chain attacks" are particularly concerning as they can have large impacts. The US government has claimed that Russian hackers supported by the Russian government were responsible. Russia's intelligence Chief has responded with the accusation that the UK and US organised the attack although it is not clear to me why governments would publicly attack their own infrastructure.



Further Reading

"Protecting privacy in practice," The Royal Society, London.

"Dilemmas of Privacy and Surveillance," The Royal Academy of Engineering, London, 2007.

D. Adrian, K. Bhargavan, Z. Durumeric, P. Gaudry, M. Green, J. A. Halderman, N. Heninger, D. Springall, E. Thomé, L. Valenta, B. VanderSloot, E. Wustrow, S. Zanella-Béguelin and Zimmermann, "Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15)*, New York, 2015.

[12] Somebody discovered that, among other things, the hash omitted to check for the sign of the transaction. Thus, a legitimate payment to Mt Gox of $10 could become -$10 without disturbing the hash.

[13] End-to-end encrypted messages are readable only by the sender and receiver and not by any intermediaries.

R. Castellucci, "Brainflayer GitHub archive," [Online]. Available: https://github.com/ryancdotorg/brainflayer. [Accessed May 2021].

T. Wheeler, "Cyber War Crimes," 30 March 2021. [Online]. Available: https://www.gresham.ac.uk/lectures-and-events/cyberwar-crimes. [Accessed 18 May 2021].

M. Thomas, "The Internet of Things," Gresham College, 20 March 2018. [Online]. Available: https://www.gresham.ac.uk/lectures-and-events/the-internet-of-things. [Accessed 12 April 2021].