Input to the
Commission on Enhancing National Cybersecurity

on behalf of
Integrated InfoSec, United Kingdom

6th September 2016

Contents

1. Submitting Parties

Integrated InfoSec is a consultancy based in the UK, specialising in the management of information risk and standards compliance, with core practices in corporate and personal privacy and the security and compliance of payment systems.

Michael Barwise (Director, Integrated InfoSec and corresponding author) holds a first class BSc degree with a computer science major, is a Chartered Engineer and a professional member of both the BCS and the IISP. He has over 30 years experience as a systems architect and engineer, technical educator and information risk consultant. Michael has contributed to national and international cyber security standards and consultations and is a member of the UK ISO 27000 standards expert panel.

Ian Stewart (Consultant) holds a Masters degree in computing for commerce and industry. He has over 40 years experience in mainframe, microcomputer and web software development, and has previously contributed to national cyber security consultations.

2. Disclaimer

The examples cited in this response have been selected solely as generically illustrative of issues relevant to the matter under discussion. They should not be construed as adverse comment on any corporation, agency, legal or natural person to which they may directly, indirectly or by implication refer, but merely as expressions of honest professional opinion relating to the issues they illustrate.

For brevity and readability we have avoided duplicating technical detail where it is available from the sources we cite. We recommend those interested in the full technical detail to study the cited sources in parallel with this response. All cited materials are readily downloadable from the public web at the time of writing.

3. Executive Summary

Although this RFI is dedicated the cybersecurity of the USA, we start from the premise that the core principles that underpin it are universal in both nature and application. In ‘cyber space’ the security of any individual nation depends on the security of all the nations with which it connects. Legislative and regulatory solutions in particular need to be transnational to be truly effective.

We contend that the widely held view of cybersecurity as essentially an adversarial domain of attack and defence is too narrow for safety. In an increasingly interconnected world, systems failure can be at least, if not more, damaging, and can originate from the same root causes as vulnerability to attack. As a consequence of innovations such as the Internet of Things, smart cities, autonomous vehicles and ‘online government’, we contend that the scope of the requirement for cybersecurity is also much wider culturally, technologically and politically than is commonly considered.

If the currently poor state of cybersecurity is to be improved, the need is for pre-emptive resilience derived from adequate practice at all phases of the system life cycle, rather than continued reliance on intrinsically vulnerable systems and reactive countermeasures as at present. However that will require significant readjustment of the way digital systems are created and managed. We assert that in the digital domain current practice at all phases of this life cycle (with the probable exception of silicon design) fails to attain the quality taken for granted in general engineering disciplines, and that this inadequacy of performance is the real proximate source of the increasingly self evident fragility of digital systems.

What primarily saddens us is that poor cybersecurity continues to grow in scale and significance, rather than being progressively brought under control as one might legitimately expect of a problem which has existed for so long with such far reaching consequences. This suggests that rather than merely continuing to do, however assiduously, what has so far failed to deliver, new approaches must be sought, and these must address root causes rather than mere symptoms.

To raise digital systems to accepted engineering standards, education and examination in the skills required at all phases of the system life cycle must be improved, and security from the ground up must become a cultural and commercial priority throughout that life cycle.

The required educational changes will have to be much wider than merely addressing technical expertise in digital systems. Fundamentally, education must move from primarily instilling facts to primarily inculcating and encouraging effective perception, attention and forethought, as these are what lead to the understanding that differentiates excellence from mediocrity in engineering. Education of this nature should be available to the public at large, not just to those already aiming at ‘digital’ careers, as informed public demand can be a strong force for improvement.

Direct incentives will also probably be necessary, although they are unlikely to be really effective until the effects of improved education have taken hold, as until then the capacities to support their aims will not be sufficiently prevalent.

Positive incentives could include preferential access to government contracts for those demonstrating adequate attention to security as at present, but the standards against which security is validated for that purpose will have to become much more stringent and relevant than they currently seem to be. To engage with market forces, it would also be worth considering voluntary security certification of vendor offerings by dedicated national bodies to common, preferably internationally defined, standards. This could be seen by the marketplace as offering commercial advantage provided the demand for secure products and services is already in place.

Negative incentives must, we believe, ultimately include mandatory liability for software defects, and the replacement of direct financial penalties for actionable breaches by nationally standardised formal audit and enforced and verified remediation driving actual improvement, albeit locally in any given instance.

Finally, we consider that improvements in cyber security can not be left to the market but must be driven by government policy, and that short term narrowly technocentric initiatives are unlikely to make a sufficient contribution to improving the overall state of cyber security to be objectively worthwhile.

4. Our position

What are we discussing? The NATO Cooperative Defence Centre of Excellence notes that "*There are no common definitions for Cyber terms - they are understood to mean different things by different nations/organisations, despite prevalence in mainstream media and in national and international organisational statements*",[1] and this is borne out by the plethora of disparate formal and informal definitions returned by web searches we have conducted. However there is a strong general (but, we believe, short sighted) tendency for definitions to emphasise adversarial causalities ('hacking'; 'cyber attack') at the expense of systems failure, and most definitions are technocentric, making little or no reference to either wider consequences or causal factors that are not directly 'digital'.

We consider that both the implications and causes of cybersecurity are much more far reaching than this. We therefore interpret cybersecurity for the purposes of this RFI in its widest sense – as **a state of robustness against both attack and failure of any system that relies on digital technologies, and against the consequential effects of such attack or failure**. For brevity we may refer to this henceforward simply as security and its antithesis as insecurity.

Even in the adversarial space, we assert that deficiencies in systems and processes are at least as significant as root causes of insecurity as the activities of attackers against them. In our experience most breaches initially reported as resulting from 'sophisticated attacks' turn out on examination to have been significantly facilitated by basic errors or omissions on the part of the target or members of its supply chain. We further assert that deficiencies arising from the same root causes can also contribute to, sometimes catastrophic, system failures in the absence of any attack. The primary common characteristic of these deficiencies – that of resulting from inadequate attention, knowledge or process rather than being mere outcomes of bad luck – tends to be obscured by the general use of the term 'vulnerability' to encapsulate them all regardless of their specifics.

Vulnerabilities can be technological (e.g. software bugs or inadequate firewall rules) or non-technological. The latter are typified by failings in decisions, processes and procedures, and they are frequently the unrecognised causes of identified technological vulnerabilities. Nevertheless, non-technological vulnerabilities are typically less frequently addressed than technological vulnerabilities, which are themselves more commonly repaired after the fact rather than actively addressed pre-emptively. This practice is so culturally ingrained that it is now taken for granted as a key security control ('updates', 'patches' or 'workarounds'). But it is really not a control at all, any more than plastering over the cracks that develop in the walls of a house with failed foundations will prevent the ultimate collapse of the structure.

To escape from this fragile reactive state, the creation and application of digital systems must rise to a level of maturity that consistently delivers results that are secure to start with – a state now taken for granted in fields such as civil, mechanical and electrical engineering, and generally attained by them. We maintain that there is no intrinsic and insurmountable obstacle to digital systems achieving comparable engineering standards, but to do so will require radical revision of the ways their creation and application are performed. In the early mainframe computing days systems design, implementation and management aspired to common engineering standards, but the rise of the microcomputer broke the mould by being largely driven from garages and back bedrooms by self-taught enthusiasts – the original 'hackers', among whom one of us was numbered. Most of these innovators learned by trial and error, making up their own rules as they went along, and their successors and descendents still tend to do the same. That is a strongly ingrained component of the digital systems culture, but it has to change as it is the antithesis of the mind set required for robust engineering.

So what is the required mind set? Although the creation of intrinsically robust systems is frequently referred to as 'security by design', we suggest that the term does not encompass the totality of the obligation, as design is only one component of the system life cycle. It is a truism that good security is generally reliant on 'defence in depth' – multiple independent layers of protection that must all fail for system failure to occur. 'Depth' is therefore commonly interpreted as multiple concurrent layers of technologies. While we concur that these may be necessary, we do not consider them sufficient. Defence in depth requires a temporal dimension as well. Security must be addressed at all points on the system life cycle – concept, design, implementation, deployment, operation, and possibly even decommissioning.

[1] https://ccdcoe.org/cyber-definitions.html

4.1. Summary

The security of a system throughout its entire life cycle depends on factors including, among others, the standards, education and training, process management, tools and methodologies, that are brought to bear at each phase of that life cycle. Furthermore, for each of those factors to be of adequate quality to support the lifecycle security of systems, security principles must be considered at all points on its own life cycle. Secure systems cannot be created using standards, processes and tools that are themselves created or used without adequate consideration of security. We therefore assert that improved security is not achieved solely or even primarily by attention merely to technical aspects of the target system during its creation, but by applying an appropriate culture and mind set (awareness of risks, attention to detail and concern for quality) to the entire system life cycle and to the processes and tools that are employed at all its phases.

5. How bad is the status quo?

The examples discussed here can only be a minute sample of real-world instances, and that sample is not necessarily representative of the distribution of the population of all incidents. They have been selected as both relatively current and clearly illustrative of the issues we consider important and because they are sufficiently well documented to be considered as reliable evidence.

Starting at the consumer end, we find remote controlled domestic intruder alarms[2] that can be disabled or bypassed by intercepting and replaying their wireless control signals because the communication channel they use is not encrypted. Such devices are used to protect millions of homes, so the potential societal harm could be significant. The user is unlikely to be aware of their vulnerability, but it might well invalidate their insurance, resulting in unrecoverable losses.

Remotely controlled access devices ranging from locks used for securing bicycles and gun cabinets[3] to centralised physical access controllers used in government buildings and airports[4] have been found to contain similar vulnerabilities to those found in alarms. The consequences could range from robbery to terrorism.

Surveillance equipment – CCTV and digital video recorders (DVRs); is now notorious for leaking video streams onto the public internet.[5] Indeed two web sites[6] now specialise in displaying feeds from such systems and too many of these feeds for comfort display sensitive information, including, in one case we noted, views of the machine room of a data centre.

Even more prevalent are vulnerabilities in networking appliances. Small and home office (SOHO) routers are probably the most common endpoints on the internet and are used, not just in homes, but by the smaller members of the global supply chain. Many vulnerabilities in these routers allow them to be remotely reconfigured,[7] which could result in security breach of the private networks behind them.

SOHO routers are the most commonly cited offenders, but there are also instances of commercial and industrial grade network devices that contain very basic exploitable vulnerabilities that facilitate security breaches, such as hard coded encryption keys common to all devices of the same model.[8] Due to the interconnectivity of the supply chain, consequences could range from compromise of individual businesses to cascade breaches affecting major corporations or government agencies.

So far we have looked at physical digital devices, but we find equally worrying vulnerabilities in mainstream digital services. The February 2012 eight hour failure of the Microsoft Windows Azure[9] service seems ultimately to have been due to a conceptual error on the part of a software developer. The consequences of a single poor decision lay dormant until specific unpredicted circumstances enabled it to trigger the collapse of an entire world-wide service.

[2] http://blog.ioactive.com/2016/02/remotely-disabling-wireless-burglar.html

[3] https://media.defcon.org/DEF%20CON%2024/DEF%20CON%2024%20presentations/DEFCON-24-Rose-Ramsey-Picking-Bluetooth-Low-Energy-Locks.pdf

[4] http://blog.trendmicro.com/let-get-door-remote-root-vulnerability-hid-door-controllers/

[5] https://www.pentestpartners.com/blog/pwning-cctv-cameras/

[6] www.insecam.org shodan.io

[7] http://seclists.org/fulldisclosure/2015/May/129

[8] https://ics-cert.us-cert.gov/advisories/ICSA-15-160-01

[9] https://azure.microsoft.com/en-us/blog/summary-of-windows-azure-service-disruption-on-feb-29th-2012/

More significant have been breaches of trusted agencies such as Certification Authorities[10] that facilitated the fraudulent creation of encryption certificates used for secure internet communication. In the example we cite (Diginotar), security controls existed but there were multiple failures in their implementation and management, the cumulative effect of which was inadequate resilience.[11] The secondary consequences of such breaches can be extremely far reaching as they jeopardise the security of all parties that rely on the fraudulent certificates.

We make no apology for including in the category of 'cyber incident' the destruction of the Northgate Information Solutions commercial data centre at Buncefield, Hertfordshire, UK[12] in 2005. A gasoline vapour cloud at the fuel depot sited some few hundred yards from the data centre exploded, creating a blast described as the largest in Europe since WW2. Although the causes of the blast were not (with a couple of minor exceptions) in the digital domain, the consequences for the affected parties, including a hospital, a police force and many commercial enterprises, most definitely were – the data centre on which they relied was gutted and their data processing halted.

We have to question the (hypothetical) risk assessment that led to a commercial data centre being sited next to a long-established gasoline storage facility. Indeed the gradual encroachment of real estate development into what was originally a protection zone of open land surrounding the facility takes some justification, but this is actually quite commonplace, at least in the UK.[13] The UK Health and Safety Executive's land use planning methodology[14] currently works on the basis that "*the risk considered is the residual risk which remains after all reasonably practicable preventative measures have been taken to ensure compliance with the requirements of the Health and Safety at Work etc. Act 1974 and its relevant statutory provisions*", which assumes everything is always working perfectly, making no allowance for the well recognised effects of normalised deviance on operational processes. This was a dominant contributor to the Buncefield incident, but the primary relevance of this incident is that it highlights how far back – indeed to the very earliest stages in the life cycle – deficiencies in the mindset applied can contribute to insecurity.

On a lighter note (in retrospect anyway), in March 2011 Armenia was disconnected from the global internet for several hours by a 75 year old Georgian pensioner who accidentally cut through fibre optic cables near Tbilisi while scavenging for copper.[15] The sole fibre links to each of the three Armenian wholesale ISPs apparently ran in the same duct. Numerous similar incidents world wide have disrupted internet connectivity, although to date mostly with more local consequences. But even where resilience is actively considered, insufficient attention to the bigger picture may render it ineffective. One of us is aware of an unpublished incident of this kind in the UK. A commercial ISP serving banks among others took the trouble to engage both a primary and an independent fallback wholesale connectivity provider, unaware that the trunk fibres from themselves to both ran in the same highway duct. When vandals set fire to the duct, both services were simultaneously lost.

Vulnerabilities to both accident and attack may exist throughout non-digital infrastructure that nevertheless fundamentally underpins the digital, so these examples illustrate the need for a continuum of security from the physical to the digital. The same interrelation can also be seen in the reverse direction. In January 2015 the Federal Aviation Authority issued an Airworthiness Directive on the Boeing 787 airliner.[16] It appears that its onboard electrical generators, essential for maintaining flight capability, would stop supplying power after a fixed period, by implication independent of whether the airplane was in flight or not. The cause was apparently the overflowing of a software counter in the generator control units. Somewhat counter-intuitively, the FAA directive describes the condition as "*going into failsafe mode*" – the inverse of the potential reality for the airplane as a whole.

Fortunately the 787 generator vulnerability was found before it precipitated an accident, but that was not the so in the case of an Airbus A400M military transport that crashed with fatalities near Seville, Spain in

[10] https://www.rijksoverheid.nl/binaries/rijksoverheid/documenten/rapporten/2012/08/13/black-tulip-update/black-tulip-update.pdf

[11] http://intinfosec.com/library/incidents/2011-How_Not_to_Secure_a_CA.pdf

[12] http://www.hse.gov.uk/comah/buncefield/buncefield-report.pdf

[13] http://www.ukopa.co.uk/pdfs/UKOPA-08-0063.pdf

[14] http://www.hse.gov.uk/landuseplanning/methodology.htm

[15] https://www.theguardian.com/world/2011/apr/06/georgian-woman-cuts-web-access

[16] https://www.gpo.gov/fdsys/pkg/FR-2015-05-01/pdf/2015-10066.pdf

May 2015.[17,18] A software install apparently deleted parameters in three of the four engine control units, causing the engines to fail to respond as expected to the throttle controls after takeoff. We wonder why the engines were apparently by design able to start and the plane to leave the ground in the absence of the required parameters.

Finally, in terms of sheer numbers of instances, we find insecure practices ingrained in web application development dominating the online vulnerability landscape. While the consequences of most of these may be relatively slight individually, their prevalence presents a vast attack surface via which other systems may be breached in cascade. Most notable is the almost universal use of parameters in URLs (the HTTP GET method) for communication and control between a web application client and its server. The fundamental problem is that the parameters can be tampered with on the client side, so GET should never be used where this may cause unwanted behaviour. A classic example is manipulation of an account number or other credential among the URL parameters, granting access to someone else's account on the server side. The expertise required to do this is virtually nil. Another method for communicating with a server exists (HTTP POST), which is much more difficult to abuse from the client side and which is designed for exchanges where such abuse might be damaging. Unfortunately, the internet Request For Comments (RFC 7231)[19] that defines both methods uses language that is potentially misleading to the general reader, describing GET as "*the primary mechanism of information retrieval*" and defining it as "*safe*". Although this implies to the uninitiated that it is 'safe to use', the RFC actually defines safe elsewhere in the text as only to be used when it is safe to do so – when "*the client does not request, and does not expect, any state change on the origin server*". Consequent on this idiosyncratic usage, and in view of its simplicity of implementation, GET is commonly used where it is *not* safe to do so.

5.1. Summary

We find that regardless of the technical specifics, hazardous vulnerabilities are rife in pretty much every application of digital technologies, and the most common contributory factors are: overlooking the bigger picture (particularly the operational context); errors or omissions in one or more phases of the life cycle, either due to inadequate conceptualisation or attention, or for reasons of expediency; inadequate, or omission of, active verification of security.

6. Sources of insecurity

The vulnerabilities we have identified arise directly from the way digital systems are conceived, designed, deployed and managed, but that ultimately depends on the thought processes brought to bear at all those stages. What all the above apparently disparate examples have in common is that the individual vulnerabilities are per se not esoteric but result from quite basic errors, wherever they may be perpetrated in the life cycle. So let's review those life cycle stages.

6.1. Conceptual and design errors

Conceptual errors can be fundamental – why unlock a gun cabinet or bicycle from 30 yards away? However commercial forces can and often do also lead to conceptual errors. The 'internet of things' is littered with examples of devices that offer little or no advantage to the user by being connected via the internet, but the option of monetising an otherwise unprofitable product by offering a supporting web service may turn it into a commercially viable product. Given that the commercial benefit is in the service and the margins for the device are small, sufficient attention may not be given to its vulnerabilities. For example the remote controlled lock may be the most attractive commercial option regardless of its questionable benefits, if only because the electronics are cheaper to manufacture in bulk than mechanical parts. In passing we note that almost all automotive locking has been performed remotely for quite some time, despite offering little or no real advantage to the driver and potentially facilitating auto crime.[20]

Conceptual errors can also arise from concentration on a single aspect of the problem in hand at the expense of the bigger picture. This is particularly likely where a digital system is intrinsically complicated or is only a small component in a larger engineered product or facility. We hypothesise that the A400M engine control units were most likely at least partially dedicated to avoiding engine damage, and preventing overspeed was key to that objective as conceived. Consequently loss of control in flight

[17] http://www.bbc.co.uk/news/technology-33078767

[18] http://aviation-safety.net/database/record.php?id=20150509-0

[19] https://tools.ietf.org/html/rfc7231

[20] https://www.cs.bham.ac.uk/~garciaf/publications/lock_it_and_still_lose_it.pdf

resulting from the way overspeed was prevented (by overriding the flight deck throttle controls) was overlooked, it not being anticipated that three of the four engines would be affected simultaneously.

Design errors occur at the point where the concept is translated into the technical specification of a realisable product or service. At this point economics typically loom large in the decision process. Fundamental flaws such as omitting encryption from a communication channel may be dictated by commercial considerations rather than by any error of technical judgement. The increased cost of using a more powerful processor or more memory needed to support encryption may tip the balance between a saleable product and an uncompetitative one. However design errors also occur due to poor application of engineering principles in the absence of such considerations. The notorious ‘heartbleed’ vulnerability[21] in the OpenSSL software library was widely ascribed to an implementation error in that library, but examination of the Request For Comments (RFC 6520) in which the offending ‘heartbeat extension’ was defined[22] leads us to consider that it has a fundamental design flaw in that is over-complicated for the task. It unnecessarily allows messages of arbitrary length to be sent and received, where, as far as we can determine, a fixed length message (or even a fixed content message) could have served the required purpose. The RFC merely states “*If the payload_length of a received HeartbeatMessage is too large, the received HeartbeatMessage MUST be discarded silently*” which places the onus on the implementer, and significantly the implementer of the victim side subject to the heartbleed attack, to cover for the design flaw.

Conceptual and design errors occur prior to what is commonly considered ‘software development’ but they may also contribute to the emergence of vulnerabilities during that process and at later stages in the system life cycle, as early decisions may influence or even restrict later options. Therefore, as in general engineering practice, we consider concept and design as fundamental parts of the development process.

6.2. Implementation errors

Implementation errors dominate numerically as a source of software insecurity. They range from machine level vulnerabilities such as buffer overflows[23] and pointer mismanagement[24] to high level errors including performing critical and security-related processes using client side scripts which can be tampered with by an attacker. In one such case (a web enabled DVR)[25] user authentication was performed using client side scripting and authentication success was indicated by the script setting three web cookies. Setting these cookies manually in the web browser granted access to the DVR independent of authentication via the script.

An excessively common and extremely dangerous implementation error is failure to validate user-supplied data before passing it to a server for processing. This can allow anything from denial of service to complete system compromise. For example, a high security physical access controller could be forced by illicit commands to unlock the entire set of locks to which it was connected.[26]

Our experience suggests that implementation errors in software commonly stem from a culture of over-reliance on third party development system frameworks and code libraries rather than on independent thinking on the part of the developer. On the one hand, library methods may be shoehorned into performing tasks for which they are not entirely suited, and on the other, tasks may be modified to fit them to the capabilities of available library methods, resulting in changes to functionality or the introduction of fragility. Where neither is possible, code written in substitute is likely to be of questionable quality due to a lack of individual expertise dictated by the dominant cultural reliance on frameworks and libraries. Alternatively, a design requirement may even be dropped at the implementation stage. On a major SaaS project, one of us found that the development team had silently abandoned the user input validation requirements we had specified, which needed bespoke coding, in favour of using a generic library method that did not exercise the same level of rigour.

The use of third party tools, frameworks and code libraries is strongly driven by economics. It is vastly more productive in terms of code volume per working day to lace together calls to extant library routines than to code the same functionality independently. It shortens product release cycles and at best it also contributes to interoperability by standardising code. However excessive reliance on it has less tangible

[21] http://heartbleed.com/

[22] https://tools.ietf.org/html/rfc6520

[23] https://www.owasp.org/index.php/Buffer_Overflow

[24] http://blogs.cisco.com/security/talos/exploiting-use-after-free

[25] https://www.pentestpartners.com/blog/pwning-cctv-cameras/

[26] http://blog.trendmicro.com/let-get-door-remote-root-vulnerability-hid-door-controllers/

but important negative consequences. Firstly, as in the case of 'heartbleed', if a vulnerability exists in a library it will be present in all code that uses it (and that is more common than is widely acknowledged). Secondly, it abstracts the development task, inhibiting understanding of the relationship between what is entered at the keyboard and what will be executed by the final system, which makes self-monitoring of code quality effectively impossible. Consequently it fosters and perpetuates 'dashboard knowledge'[27] – the ability to manipulate the development tools without understanding how they work or what the outcomes are of their use.

Increasing automation of the development process can also be a contributor to implementation flaws. The use of 'drag and drop' graphical development tools abstracts the process even further, to such an extent that many practicing software developers, particularly in web development, seem to have problems understanding the code of fundamental functions – witness the vast number of 'crowd sourced' code fragment sites on the web.[28] Whatever the quality of the code therein, the user comments on such sites commonly exhibit lack of even the most basic understanding of the process the code is required to perform or how the code accomplishes it.

At the other end of the implementation process, automated code optimisation can introduce vulnerabilities that are not apparent in the source code, for example by selectively omitting code statements that the compiler considers improper or redundant according to the strict rules of the programming language.[29] This is particularly problematic where the source code takes even small liberties with the language specification – a common occurrence in the case of code written under commercial pressure by inexperienced developers. As modern applications can run to millions of lines of source code it is mostly infeasible to examine the executable code, so changes introduced by automated optimisation of imperfect source code can contribute invisibly to insecurity by rendering the executable an unknown that differs from the expectations engendered by examination of the source code.

For all these reasons, even if source code passes review, that may be insufficient to detect vulnerabilities in the executable program – witness the veritable torrent of security patches that are now taken for granted from mainstream vendors, despite all of them having formal testing processes. Furthermore, as development systems and libraries are commercial products like any other, corrections are only made where it is cost-effective to do so, and this consideration retards any drive for improvements in quality or security.

The key adverse outcome of all this is that a small number of development systems vendors and library developers effectively dictate the quality (and thus the security) of the output of pretty much all other software developers, who use the tools largely blindly. Consequently, individual software flaws can be widely present in the population of software as a whole rather than localised in the output of individual developers. Therefore, although we cannot disapprove of proprietary development systems, libraries or automated code optimisation – they are essential in economic terms and at best contribute to consistency and interoperability – we suggest that the negatives their use encourages must be recognised and countered actively by other means.

Technological aspects such as developer expertise, code library quality and automation are major contributors to security or the lack of it in the implementation phase, but over-arching development management processes and methodologies are also very significant. We have so far looked at each phase of the system creation process individually, but methodologies used throughout the entire process (from concept through design to implementation) can have significant effects for good or evil. For example the now widely adopted Agile methodology speeds up software development but makes security hard to ensure.

Based on individual requirements described in 'user stories', code is written in modules which are cyclically revised until the 'user' passes the module as fulfilling their stated requirement. In practice, user stories tend to be extremely brief and superficial – 'I want to be able to do X'.

In engineering terms, this approach effectively eliminates the design stage. The user story is a concept, and that is passed straight to the implementation of code. Consequently, recognition and addressing of any side effects or potential vulnerabilities depends on the expertise, perception and attention to detail of

[27] http://davidlavery.net/barfield/encyclopedia_barfieldiana/lexicon/dashboard.html

[28] e.g. http://www.codeproject.com/

[29] https://pdos.csail.mit.edu/papers/stack:sosp13.pdf

the individual coder, as it is very unlikely that a user story is specified in enough detail to ensure these are covered in the brief. In most cases the 'user' is unlikely to be aware of the need, but even a user story such as 'I want to be able to do X securely' leaves open the question 'securely against what?' So much is left to the discretion of the coder that unless rigorous internal standards are established for, and followed consistently by, the entire development team the resulting system will likely exhibit unpredictable and inconsistent security. Such internal standards are at present a rarity rather than the norm, but even if they exist they may be seen as interfering with the prime purpose of the agile methodology – rapid code development.

6.3. Deployment errors

The security of otherwise adequate systems can be, and often is, compromised at the deployment stage. In respect of software, deployment errors commonly include misconfiguration and using obsolete versions that are no longer supported by the vendor, both of which can render the system vulnerable.

Deployment vulnerabilities frequently result from much wider considerations than just software. This is commonly encountered in commercial networks supporting card payment systems. A certified payment management system gets connected to certified card readers across a flat network that carries other business traffic and has endpoints connected to the public internet, thereby exposing payment card data to possible breach. The TJX breach of 2005[30] was a prime example of this. Some 45 million sets of card data were exfiltrated via insecure wireless networks over a period of about 18 months. At that time the TJX breach was the largest of its kind on record, but it has since been displaced by others. The problem is growing, despite there existing a very explicit and prescriptive standard (PCI DSS) for the secure management of card payment infrastructures.

Verizon, who perform forensics after payment card breaches, stated in their 2015 PCI compliance report[31] that "*Of all the companies investigated by our forensics team over the last 10 years following a breach, not one was found to have been fully PCI DSS compliant at the time of the breach.*" This is supported by the experience of one of us in guiding organisations towards compliance with PCI DSS. Many organisations seem not to have studied the standard in any depth. Instead of implementing the necessary framework of permanent criteria and processes for maintaining compliance and then deploying systems using those criteria and processes, they opt for ad hoc deployment of a set of independent technical fixes based on a cursory reading of each clause of the standard in isolation.

This perfunctory approach is representative of common failings that lead to insecurity, and the key component of these failings is cultural. In order for the resources to be allocated to compliance with any externally imposed standard, compliance has to matter enough to justify those resources. At present it often doesn't, as it is not considered to affect the bottom line. Certification of compliance may be a prerequisite for access to resources, connectivity or contracts, but businesses may be allowed to self-audit their compliance without any external verification of those audits unless a breach occurs. Consequently, some standards that could potentially drive the rigorous application of basic good security practice, and which could with advantage be applied in principle across the entire corporate infrastructure, are widely treated as narrow 'compliance obligations' entirely divorced from 'security', to be fulfilled in name only with the least expenditure of effort, the root cause being lack of motivation.

Another important source of insecurity at the deployment phase is the abysmal quality of much systems documentation. Where formal documentation still exists (and it is increasingly a rarity) bizarre translations, ambiguities and omissions of detail abound. Admittedly at the budget end of the scale, a consumer DVR manual[32] we have seen, contains, among other gems, the statement "*If the sound box and the tone arm can not be isolated, howling phenomena is often existed*." In addition, the manual covers seven different models of DVR without identifying which of many functions it cursorily describes pertain to which model, and some of the controls present in the user interface of one model are not mentioned in the manual at all. It is not surprising to us that such DVRs get misconfigured, all too often leaving them vulnerable. But the problem of poor documentation is far from exclusive to consumer systems. There is a growing tendency for 'user communities' and 'wikis' to serve as the primary sources of user support even for high end systems such as commercial grade software development tools and even operating systems. This must surely contribute to and perpetuate misconceptions and poor practice. It is widely recognised that

[30] http://www.computerworld.com/article/2538711/cybercrime-hacking/one-year-later--five-takeaways-from-the-tjx-breach.html

[31] http://www.verizonenterprise.com/resources/report/rp_pci-report-2015_en_xg.pdf

[32] http://www.henrys.co.uk/cctv/AHD+DVR.htm

the content of Wikipedia is not to be entirely trusted[33], expressly because anyone can contribute, and they do whether they are properly informed or not. Authoritative documentation is essential. Those who deploy digital (or any other) systems cannot be expected to perform to high standards when the guidance they have access to does not exhibit adequate levels of accuracy, clarity and completeness. We have encountered instances of sub-optimal implementation of network security appliances at the corporate level that can be traced back to this.

6.4. Summary

We find that economics and lack of incentives are key contributors to failure to achieve security at most phases of the life cycle, but that the wide acceptability of low levels of professional expertise at any of its phases is also a significant factor. It is entirely possible that the two are interdependent, with economics the driving factor.

7. Common factors

In taking stock of the apparently disparate failings we have discussed so far, we must identify their most significant common factors across the entire system life cycle. We consider these to be primarily conceptual and cultural rather than technical.

7.1 Failings of standards

To avoid conflicts of interest, we intentionally do not identify any specific standards in this section of our response.

Standards should ideally provide a high quality, consistent framework of trustworthy guidance that allows those who follow them to achieve adequate performance in the domain of the standard. At present this aim is far from fully achieved.

Multiple standards relevant to, or which make reference to, security exist, originated by different bodies. Some of these directly address security, but many are more general – dealing, for example, with IT management or service delivery but including requirements or guidance on processes that relate to security, such as risk assessment. Although they may have the same intent and even describe the same requirements or processes, parallel standards tend to use their own idiosyncratic vocabulary, are effectively in competition with each other, and each is tacitly assumed by its originators and advocates to be sufficient of itself. Consequently the user community tends to fixate on compliance with the letter of their chosen standard instead of considering the spirit and intent common to them all. This is exacerbated for those standards where certification of compliance is an option, as gaining certification tends to take precedence over any real world results of adoption, both for users and auditors. Literal nonconformities with the specific wording of clauses in the standard become more important to all concerned than success or failure in achieving what those clauses are intended to ensure happens.

Another common failing of many standards is that the guidance offered is mostly extremely general, omitting detailed discussion of the way the standard's requirements can be realised. Even where fulfilment of requirements is discussed, a standard may concentrate on administrative processes rather than on how to achieve outcomes of high quality. Thus one may be required to ensure that risk assessment is repeatable and reliable, but how to achieve and verify these requirements, which is the hard part of the problem, is not explained other than perhaps in terms of procedures. In the absence of the necessary guidance, both adopters and auditors tend to settle at best for repeatable and reliable processes, rather than what is intended – repeatable and reliable results.

Were standards to refer to first principles, however, they might indeed support improvement. For example, a small number of simple first principles underpin the determination of likelihood (and are thus essential to good quality risk analysis). These principles have been established and recognised as valid for almost 400 years, but no information security or risk management standard we have seen makes reference to them. Instead, those standards merely reiterate flawed current practice, which, in the case of risk analysis for example, includes fundamental misinformation that widely leads to meaningless results.[34]

For all these reasons we consider standards in their current guise as necessary, but not sufficient to drive improvement in security. They can inform the uninformed of the absolute basics but cannot generally guide the informed towards improved performance. However many adopters erroneously consider even

[33] http://www.instituteforpr.org/perceptions-wikipedia-public-relations-professionals-comparison-2012-2013-surveys/

[34] https://my.nps.edu/documents/103424423/106950799/DRMI+Working+Paper+2011-2.pdf

standards that are self-designated as baseline to be ‘gold’ standards, causing them to terminate their compliance efforts immediately on fulfilling the minimum requirements of their standard of choice. The marketing and promotion of standards by both standards bodies and third parties such as auditors and consultancies is largely to blame for this, as is a real shortage of generally applicable true gold standards.

In summary, we consider the primary root causes of the currently limited utility of standards in supporting security to be economic and cultural. The key economic factor is that standards have to be saleable to the widest body of adopters, who tend to seek the lowest cost of compliance, and thus prefer minimally prescriptive and demanding standards. The key cultural factor is that the content of standards is generally dictated by established practitioners who by definition subscribe to the consensus of that current practice, causing it to become entrenched against change even where it is deficient.

7.2. Failings of education

The fundamental failings of current education in the security domain – indeed in the entire digital technology domain with the exception of silicon design – are merely representative of the fundamental failings of current education in general. A strong emphasis on rote factual memorisation at the expense of understanding of underlying principles has been evident for several decades throughout the general educational system in most economically advanced nations. This has created from early years learning onward a defective way of relating to knowledge that has nevertheless become accepted as a norm. It is exemplified in the digital domain by the prevalence of short courses in elementary coding that masquerade as professional developer education.

In August 2016 the US Department of Education published a fact sheet on a potentially very worthwhile initiative to grant access to education to low income students.[35] However the offerings in software development it describes, two out of three of which are of 12 and 13 weeks duration respectively, cause us significant concern. Our greatest single objection is that the 13 week course in web development describes its outcome as preparing students “*for jobs as mid-level software engineers.*” This is a completely unrealistic expectation if the word ‘engineer’ is interpreted in its normal professional sense. It takes from three to seven years of intensive post-high school education to train an entrant to any of the established engineering professions and around five years post-education practice to reach mid-level rank.

We consider that a short course of 13 weeks is only capable of familiarising students with the most basic generic programming structures (e.g. loops and branches), the ‘knobs and levers’ of a vendor-specific development system and the use, but not the mechanisms, of a limited subset of libraries. It cannot teach how to code except mechanically, and cannot possibly impart even the most elementary skills in defensive programming. Nevertheless, in the community at present, knowing the IDE and libraries equals ‘developer’, so the fault most likely lies primarily not with the definers of this course but with the culture within they have been obliged to define it.

The adverse outcome of such offerings is self evident. The top four vulnerabilities in the OWASP Top Ten[36] prevalence ranking of web application vulnerabilities, have shuffled places from time to time but have communally remained at the top of the list for over a decade despite being widely discussed, and supposedly widely understood, by the developer community.

We also take issue with the drives of several national administrations to introduce ‘coding’ into the mandatory educational curriculum from early years. To redress a shortage of expertise for delivering this, there have been moves, at least in the UK, to provide crash courses in coding for existing teachers[37,38] so they can deliver the curriculum without prior experience in the subject. The likely outcome is merely to release further echelons of inadequate software developers into an already broken profession.

Software development is not the only discipline which suffers from inadequate security education. We also have major concerns about the education of those directly tasked with advising on, implementing or managing, information or systems security.

The dominant mode of education for technical security practitioners is short courses and computer

[35] http://www.ed.gov/news/press-releases/fact-sheet-ed-launches-initiative-low-income-students-access-new-generation-higher-education-providers

[36] https://www.owasp.org/index.php/Top_10_2013-Top_10

[37] https://www.gov.uk/government/news/year-of-code-and-500000-fund-to-inspire-future-tech-experts-launched

[38] e.g. https://www.codeclubpro.org/faq

marked multiple choice tests. A huge range of such training and testing exists, which in our experience neither has significant depth nor does it verify individual expertise. The content of most such courses in the security and risk spaces is either narrowly prescribed by the assumed level of an examination (rather than, as it should be, the other way round) or dedicated to a vendor product, and we have not yet found any such qualification that can be described objectively other than as entry-level.

Although vendor independent in the strict sense that they are generally offered by dedicated certifying bodies without a product line other than the qualifications themselves, we do not rank most qualifications in security operations and management much higher, For example, we have in our possession a self described 'advanced' preparation guide for a highly reputed security management practitioner examination, a pass in which is considered by many to demonstrate the epitome of security expertise. However the content of this guide consists mostly of questions on basic terminology, plus a few problems on obscure topics such as the internals of encryption algorithms – information that is unlikely to be relevant to mainstream security management practice.

Because the actual content of such examinations is commonly a contractual secret between the examining body and each individual candidate – and indeed one certifying body contractually prohibits successful candidates from even expressing opinions about their examinations – we and others can only judge their quality from the preparation materials on public offer, so it is not possible for anyone to comment entirely authoritatively. Nevertheless our impression is that such qualifications are at worst nugatory and at best, although maybe arduous and expensive to gain, irrelevant to the professional expertise required in the real world. We have seen such qualifications described as 'demonstrating commitment', but that seems to us insufficient in the absence of validation of relevant expertise.

The deficiencies we note most likely result both from the dominance of economic considerations and the intrinsic limitations of the computer marked multiple choice testing driven by them. Computer marked tests are vastly cheaper to deliver than free form question examinations, but they pretty much have to be multiple choice due to technological limitations of automated marking. Such tests can readily and accurately verify retention of simple facts, so that has become the model for the qualifications. But a multiple choice test cannot validate the ability to work out 'what is going on' in an ambiguous situation and come up with solutions to it, although the exercise of these capacities is the primary distinction between excellent and mediocre performance in engineering, as in most other aspects of life.

The tail therefore appears to be wagging the dog, as the economics and convenience of conducting the examinations seem to have defined the testing regime at the expense of the results. The aggregate effect of this is the proliferation of qualifications that satisfy human resources departments without validating the expertise of (or identifying the lack of it in) candidates, which sets a precedent for low expectations of said expertise. However, in addition to the specific economics of conducting examinations, certifying bodies face an difficult challenge. An organisation dedicated to providing professional certifications must have a publicly acceptable pass rate in their examinations in order to attract sufficient candidates to stay in business. Therefore the examinations must not be too hard for the average student to pass. Furthermore, as such an organisation typically relies on a network of training partners it cannot afford to lose their goodwill by making the required training too costly to develop, maintain and deliver. Thus there is, if not actually a race to the bottom, at least a settling at an economic but relatively low common denominator which may not (indeed in our opinion currently does not) yield validation of adequate practitioner standards.

We would add that, at least in the UK, the bachelors' degree does not at present typically pass muster as validation of skills in security. We examined the syllabus outlines of thirty randomly selected UK bachelors' degree courses in software engineering and found that only seven listed security-related modules. None of these seven courses included more than a single compulsory security related module (5.5% of course content), and in three of the seven cases such modules were entirely optional. With one exception, all the security related modules were offered in the final year, after development mindset and habits are likely to have been established by students for better or worse. We suggest that, at the very least, basic security concepts should feature as an intrinsic component of all course modules from day one, rather than being isolated as a separate, and largely optional, advanced subject.

Academic courses specialising in information security education are, in the UK at least, available only at postgraduate level. We have two key concerns about this. The first is that course content at this level is often highly theoretical (indeed 'academic') and technological, concentrating on abstractions such as cryptographic mathematics rather than being coupled to the needs of delivering real world security. Our other concern is that only a tiny minority of security practitioners undertake postgraduate courses. These practitioners form a small elite echelon from the perspective of recruitment, but their small numbers mean that even well focused courses at this level are likely to have negligible effect on professional standards as a whole.

Although academic courses may not currently be serving the real needs of would-be security practitioners, academic research output might make significant contributions to security, but for a widespread cultural rejection of anything academic by a majority of information systems practitioners. In our experience that is particularly the case in IT operations, where the results of some of the said research could be particularly beneficial if operations staff could be prevailed upon to read it. A classic example is the management of passwords, which in the operational sphere is in general appallingly bad at addressing the human equation. However there is a large body of solid academic research relating to the psychological constraints that lead to robust or fragile password management. This cultural chasm needs to be bridged in order to help dispel a significant body of rote learned security mantras that do not reflect reality and introduce some valid new approaches into common practice.

In summary, we consider the two most significant direct contributions to failure of commercial security education offerings are superficial ‘dashboard knowledge’ training instead of the imparting of principles, and automated multiple choice testing which cannot validate expertise, only rote retention. These failings are far from exclusive to security education – they permeate many general educational systems. A key contributor, however, to their extent in security education appears once again to be economics, as offerings are inevitably driven by niche market demand. In the case of academic courses we find poor practical relevance to be a significant contributor to their limited effectiveness, although academic research results could be useful if they could be made culturally acceptable to practitioners. However, most importantly, uncritical and unresearched recognition of qualifications by name, rather than content and quality, in the marketplace (particularly at the point of hiring) allows these deficiencies to persist, and we consider that it constitutes a significant retarding influence on the improvement of security education.

7.3. Failings of methodologies and tools

As software development methodologies have evolved from ‘waterfall’ where full design specifications preceded the implementation phase to ‘agile’ methods that, as we have discussed elsewhere, can effectively eliminate the design phase, we must examine how this affects security.

Waterfall development was based on meeting a full functional specification in the way best fitted to that end, and entailed the use of formal processes with fixed validation points where conformity with requirements could be checked.. The agile approach, aiming as it is intended to at swift delivery of results and coding time efficiency, is much less structured. Its primary goal is to ‘code the concept’ as quickly and conveniently as possible. This can lead to development decisions that subvert the original intent, and its check points can be quite loosely defined. Given that security is commonly considered as an assumed attribute (via not making errors) rather than as an active process of building in resilience, it becomes at best tacit in the coding process and at worst is simply disregarded. Secure delivery therefore currently relies heavily on post-coding source code review, but the scale of modern applications (maybe millions of lines of source code) precludes exhaustive validation of all possible pathways through the code. Testing therefore typically falls back on predefined black lists of known errors that cannot address the unexpected.

It is accepted best practice to perform penetration tests on applications after coding is completed. However, although these can be of value in demonstrating that vulnerabilities exist, they cannot deliver the inverse. A ‘clean’ penetration test is not a reliable indicator of a secure system. It may just mean that a vulnerability that is present has not been discovered by the applied tests. Consequently reliance on post-development testing can not provide assurance of security to any specified level. It is supplementary to, not a substitute for, secure development practices.

The quality of development tools also significantly influences code quality, and the increasing use of highly abstracted development interfaces (e.g. drag and drop) dissociates the developer from the output of the development system, making both defensive coding and manual verification impractical. This is a significant contributor to lack of control over security, even where it is firmly on the developers’ agenda.

Although we have so far refrained from appraisal of specific technologies, we feel obliged mention what is currently one of the greatest single contributors to technical insecurity on the web, but which has effectively become standard practice. The common use of client side scripting for rendering web content in preference to using the traditional content and presentation languages of the web (HTML and CSS), is probably the most ubiquitous, and certainly the most dangerous, vulnerability to which web users are exposed. Because the combination of the web browser and the scripting technologies currently used (JavaScript, AJAX) is fundamentally insecure[39] and access to a majority of web content now requires scripting to be enabled in the browser, effectively the entire web user base is constantly exposed to security threats ranging from credentials theft[40] and online payment fraud[41] to full system compromise.[42] Indeed malicious JavaScript has for a long time been, and remains, the primary vector via which most external attacks on web facing systems are launched.

Despite voluminous evidence of the hazards, this fashion has taken hold to such an extent that in many instances we have examined in detail, simple, safe and well established HTML methods are intentionally disabled, being replaced by idiosyncratic and sometimes quite complicated scripts offering identical functionality – scripting for the sake of scripting. The outcome is that the web is widely accessed, whether for personal, corporate or government purposes, in a state that is intrinsically vulnerable to a wide range of attacks, and the high success rate of such attacks is a significant contributor to general insecurity. However, one of the most insidious barriers to correcting this situation is that the vulnerability engendered by the requirement for scripting to be enabled on the client side is an externality for site owners and developers. Even if an attack is launched from their web site, the malicious scripts used have almost certainly been injected by a third party, so appreciation of 'site security' focuses on protecting against the specific means of injection (e.g. cross site scripting)[43] rather than the intrinsically vulnerable state imposed on the public by the common, largely unnecessary and insecure requirement for scripting to be enabled by default.

We are therefore concerned about methodologies and tools on two counts. Firstly and most critically, the wide application of intrinsically insecure methodologies and technologies, particularly to the delivery of web content, constitutes a global scale hazard at all levels of society that is essentially impossible to fully defend against if access is required to that content. Secondly, even if security is taken seriously in software development, testing alone is not an adequate verification of it, but development methodologies and tools increasingly used for reasons of economics and convenience have made verification of security by other means, including verified robust coding, hard to achieve. It is therefore imperative that cultural trends of these kinds are not allowed to persist in disregarding the wider requirements of security.

7.4. Failings of management

The key failings of security management are lack of commitment and normalised deviance.

Security is for most organisations seen as an overhead with no obvious payback. Incidents are relatively infrequent for any individual organisation, and when something has not apparently happened it is difficult to justify resources to prevent it. Consequently the typical organisation merely waits for incidents and responds to them reactively rather than aiming for pre-emptive resilience. Probably the most significant root cause of this state of affairs is the absence or inadequacy of risk assessment. Very few commercial organisations have a realistic appreciation of their exposure even to adversarial attack in the digital domain. For example, when we discussed this exposure with a senior technical manager of one of our international clients, their response was "*but we don't have enemies.*" The reality for all but extremely high value targets is, however, that the vulnerable will be breached – more likely mugged at random rather than victim to planned burglary.

Normalised deviance is the process whereby gradual departure from required performance proceeds unnoticed and is accepted despite not remaining in accord with objectives. The performance of even well designed processes tends to decay as they become ingrained into 'business as usual', to the point where they may become vulnerable to abuse or error or even cease to fulfil their intended function. The Buncefield incident exhibited several independent instances of normalised deviance – some of them extremely apparent to an outsider but taken for granted by those responsible for the affected processes.[44]

Normalised deviance in technical processes is especially problematic in the production environment, as

[39] http://javascript.crockford.com/security.ppt slides 1-36

[40] e.g. http://blog.checkpoint.com/2016/02/02/ebay-platform-exposed-to-severe-vulnerability/

[41] e.g. http://www.foregenix.com/blog/magento-malware-alert-malicious-client-side-javascript

[42] https://www.trustwave.com/Resources/SpiderLabs-Blog/Angler-Takes-Malvertising-to-New-Heights/

[43] https://www.owasp.org/index.php/Cross-site_Scripting_%28XSS%29

[44] e.g. http://www.hse.gov.uk/comah/buncefield/buncefield-report.pdf paragraphs 26, 46, 51, 55

its consequences are not limited in scope to the party whose processes are deviant. They frequently escape notice until they precipitate incidents further along the supply chain. In the case of the 'heartbleed' vulnerability, an error in a code library which found its way into the firmware of some high end security appliances resulted in data breaches to organisations that deployed those appliances, potentially jeopardising the customers of the affected organisations.

Normalised deviance in the production environment can be damaging enough, but in security operations it can have devastating effects. The Manning/Wikileaks affair was fundamentally facilitated by it. Despite the strict regulatory policies supposedly controlling private use of endpoints attached to SIPRNET,[45] actual practices had become lax. Personal use was common, widespread[46] and casually accepted – "*Defense (Coombs): How was it enforced? Lim: No. You trusted people.*" [47]

Formal monitoring of processes can be used to control normalised deviance. However the quality of monitoring is itself not immune to progressive decline in performance through normalised deviance. Highly conforming cultures (e.g. national security, military, critical infrastructure) are particularly susceptible to this as monitoring fatigue can set in as a result of long periods of arduous formal monitoring activity with nothing to report. In the absence of external verification by means such as regular external auditing, normalised deviance in monitoring can proceed to the point where monitoring becomes a ritual of apparent rigour that has ceased to perform a useful function, allowing the processes supposedly monitored to decline to the point of failure.

Both lack of current risk awareness and normalised deviance frequently contribute to a false sense of immunity that significantly increases actual vulnerability.

7.5. Summary

Failings of standards, education, methodologies and tools and management at all phases of the lifecycle currently contribute to insecurity. Lack of determination, lack of attention and lack of accountability are key proximate components of it. While security remains secondary to more subjectively important considerations, whether those be commercial or political, it will not be achieved. In the digital domain, economic pressures increasingly militate against expending the necessary attention and resources to achieve even consistent, let alone high, levels of security. Therefore until economic priorities change little general improvement can be hoped for.

However the fundamental root cause of insecurity, as of most societal problems, is the way it is thought about, and that is a cultural norm largely dictated by education. The long established emphasis on rote retention of factual 'knowledge' must give way to the cultivation of effective mental capacities that can better relate to objective realities, rather than merely retaining and regurgitating old mantras. The key talents required are perception, attention and forethought, and these are currently not sufficiently in evidence.

8. Some suggested remedies

We have identified failings in standards, education, methodologies and tools, and management as contributory factors to the current state of insecurity. We suggest that education is the highest priority among these factors – not merely technological education, but general education that inculcates commitment, perception, attention and foresight. The need is, and has been for a long time, for an improved way of thinking, not just new things to think about in the old way.[48]

8.1. Practitioner education

As software already permeates most areas of global society, commerce and government, if it stops, malfunctions or is subverted the repercussions are potentially very far reaching, Furthermore, as nations move increasingly towards online interfaces between government and the population, let alone 'smart cities', autonomous vehicles and robot weaponry, those repercussions could become catastrophic. The design, development, deployment and operation of digital systems must therefore all become components of a true engineering discipline on a par with civil, electrical and mechanical engineering. This will require drastic revision of current approaches to education in its broadest sense. It is already not

[45] http://www.dhra.mil/perserec/osg/s1class/siprnet.htm

[46] http://alexaobrien.com/archives/1218 sections 14, 17, 20

[47] http://alexaobrien.com/archives/1735

[48] http://intinfosec.com/library/strategy/2011-Anticipating_Adverse_Black_Swan_Events.pdf

safe to rely, as at present, on a small elite cadre of highly talented and expert leaders and a huge pool of marginally competent foot soldiers. Everyone involved in the professional creation and management of digital systems must be fully competent to deliver robustly and securely at their point on the life cycle.

The practical changes needed are recognition by developers and others that security and robustness must be actively addressed at all points on the life cycle; a thorough grounding in first principles to overcome blind reliance on tools and 'received wisdom'; the active cultivation of attention to detail, analytical capability and forethought; and rigorous practitioner testing in all of these. We do not advocate statutory practitioner certification – indeed that is not currently a requirement in most engineering disciplines, and it does not necessarily ensure quality – but we strongly recommend at least national, but preferably international, non-proprietary standardised qualifications that can gain the credibility by their consistent high quality results to be demanded as entry requirements by global commerce and industry. The necessary infrastructure may already exist in the hands of established certifying bodies. The need is for improved content quality and more effective methods of validation.

The key practical capacities required of security professionals are the ability to think for oneself and advance the discipline rather than just mechanically replicate current practice; the ability to analyse situations reliably to their first principles and synthesise from those principles appropriate responses to them, even under pressure of time and resources; the ability to foresee consequences of courses of action and address those consequences in advance. All other expectations of a security professional ultimately derive from these three, so these should be the capacities that are primarily inculcated and validated, rather than reliance on rote memory retention of facts (or even of theories).

8.2. Public education

The public (including all relevant members of business and government) must be educated about the need for security, not, as at present, primarily in relation to changing their own behaviours, but in addition to encourage them to demand more secure products and services. This could be accomplished quite effectively by mandatory public disclosure of data breach events, although there is likely to be significant resistance to this from commercial quarters. Policy makers who may seek to encourage the adoption of online government services for reasons of economics or expediency despite the technologies as implemented not yet achieving sufficient maturity to ensure security,[49] might also be a source of resistance. Realistic public appreciation of the hazards of operating in cyber space could nevertheless be highly influential in driving demand for improvement in the security of digital systems and services.

8.3. Software standards and commercial imperatives

Due to its increasing ubiquity and criticality, it is essential that software be designed and implemented robustly rather than at present released in flawed condition and fixed reactively. It must therefore become more economic to deliver robust software than to issue patches. Government procurement could be influential, provided the security standards required are both objectively realistic and embrace the requirements of the wider community. So far, efforts in this direction seem unlikely to be really effective, as many government security frameworks are so general and high level that they can do little to drive practical improvement.[50] However, when addressing detail we find from participation in consultations in the UK that government is more likely to trawl existing standards rather than innovate, thereby impeding improvement. Therefore the parochial adoption, idiosyncratic presentation and narrow remit of multiple competing security standards needs to be addressed. Because security must know no national boundaries it can only be achieved by arriving at a common consensus to which all standards bodies and government agencies subscribe.

Our ideal would therefore be an international consensus of mandatory minimum standards of security in each relevant domain, defined in terms of objectives to be met and specific criteria for their accomplishment rather than as processes to be performed – a marked departure from the current general tendency of standards. That said, it would take time for any such standards to be implemented and properly adopted, and it is likely that the cost of creating and implementing them would be high.

In the deployment and operational phases of the life cycle, it is essential to prevent a security 'race to the bottom', particularly for the small businesses which increasingly serve as gateways to larger and more globally critical systems via the supply chain. Attempts have been made to achieve security standards for

[49] e.g. http://electionwatch.unimelb.edu.au/australia-2016/articles/software-can-affect-election-results

[50] e.g. https://www.enisa.europa.eu/publications/security-guide-for-ict-procurement/at_download/fullReport

this community, for example the UK Cyber Essentials scheme,[51] but we believe more could be done by setting higher expectations. Cyber Essentials has been in operation for just over two years at the time of writing and since October 2014 has been a mandatory requirement for many potential suppliers to government. It was intended to provide an achievable security standard for small members of the supply chain that found ISO 27001 too complicated or expensive to certify against. However it is not actually comparable to ISO 27001, which is a management process standard in the security domain, rather than a security standard per se. Cyber Essentials is entirely technocentric and at its basic level is just an externally moderated annual self assessment (not an external audit) of technological controls. Although supplemented at the advanced level by a penetration test, success in such a spot check assessment is no warranty of continuous security. However in reality neither is certification under ISO 27001, for which the required processes must be in place and must conform with the standard, but no check is made by the audit of their effectiveness.

An early attempt by one of us to have the question set of Cyber Essentials extended to include a check on the level to which the controls are continuously managed failed to gain traction, but we strongly advocate enterprise security certification that includes validation of management quality, rather than just technological spot checks or process presence audits. As the saying goes ‘you can’t control what you can’t measure,’ and what fundamentally needs controlling is the quality of continuous attention to security.

8.4. Software liability

Software has become mission critical and in some cases life-critical. It is already and will continue to be a functionally dominant component of engineered products and services. It is therefore essential that liability for defects becomes mandatory and enforceable. That software is licensed and not a good can not remain as at present a basis for denial of liability. The current End User License Agreement (EULA) is generally so stringent that it precludes claims even for lack of functionality, let alone reliability or security, and the ancillary conditions imposed by the license (such as specific jurisdiction) generally make it prohibitively costly to make a claim, not least due to the disparity in resources that the two parties can usually bring to any action.

Exclusion of liability becomes even more important as mainstream applications are being progressively moved by vendors from being locally installed and licensed by instance to online services with a subscription model based on use and ‘cloud’ services are increasingly used for the storage and processing of sensitive data. Individual failures can therefore affect a much wider user base. The currently typical ‘pro rata time in lieu’ compensation offerings are of negligible value to businesses that have had production disrupted, and offer no compensation for individuals whose sensitive records have been lost or leaked.

In summary, while we accept that it is legitimate for software to be licensed rather than sold in order for the vendor to retain their intellectual property rights, this arrangement is irrelevant to and should not preclude liability for quality, safety and security. The way forward here is inevitably changes to statute. However we expect attempts at such change to meet with considerable resistance, and the need for global legislative cooperation will present a challenge.

8.5. Breach remediation and product security

We advocate a standardised approach to enforcing liability for security incidents that addresses the actualities of individual cases and drives focused and verified remediation. At present enforcement is fragmented. For example, European regulators can impose an audit where a personal data breach has occurred, but in our experience (at least in the UK) the actions that emerge from such audits are often very general in nature and may not address the root causes of the breach. Worldwide, acquiring banks can impose audits where payment card data breaches have occurred, but these typically address only those aspects of insecurity that relate to payment card data.

A national forensic service offering a standardised audit regardless of the business category of data breach and with the expertise to both specify detailed remediation and verify its fulfilment would be highly beneficial, both as a direct driver of improvement and as a salutary reminder to pay attention to security. This would require universal statutory breach notification to be maximally effective, so it would have be driven by national policy rather than private enterprise, however committed and well intentioned. However, after the cost of its creation, such a service might well largely pay for itself from its audit charges, which would probably be higher but more productive of improvement than fines.

[51] https://www.cyberstreetwise.com/cyberessentials/

Security certification of digital services, devices and software to common, preferable internationally defined, standards by a national agency might also prove useful. Such certification need not be mandatory, but could raise product and service security via perceived commercial advantage. The BSI Kitemark[52] is a prime example of success in raising standards using this approach, as, although not a legal requirement, it is universally recognised as providing assurance of quality and safety for a wide range of products.

8.6 Summary

We have outlined several avenues for improvement of security, with some emphasis on legislative change. We believe that they all both would contribute significantly to that goal and are potentially implementable. However they will not deliver fully until adequate engineering standards are achievable, most importantly in mainstream software development, as the competence to deliver to those standards is not yet sufficiently in evidence. That will have to wait for improvements in education, which we therefore consider to be the highest priority.

9. Our final word

Although software quality has loomed large in our argument and requires much improvement, it is far from our only concern. The mind set brought to bear at all phases of the system life cycle from concept to decommissioning is the fundamental driver of security or insecurity, and it needs to be of a consistent high quality throughout. Consequently we do not expect short term technical point fix initiatives to have a significant effect in improving security. Indeed we see a hazard in excessive reliance on them. They are likely to multiply in narrow technological silos,[53] consuming resources while retarding efforts towards global improvement in security by merely scoring ‘brownie points’ for addressing individual symptoms of insecurity without addressing their root causes.

Achieving the required mind set for attaining and maintaining even a moderately consistent and adequate standard of security throughout the digital space is likely to be a long hard process, primarily involving re-education and culture change. Until security becomes an automatically accepted requirement and is baked from the ground up into all digital products and services, it can not safely be taken for granted, despite all the ‘security products’ that can be thrown into the pot. The primary need is not fancier security appliances or faster patching, but vastly fewer vulnerabilities, particularly if those appliances and patches are created using the same mind set as what they ostensibly protect.

Security must cease to be an externality throughout the entire supply chain, which implies that creating secure products and services and operating them securely must become less costly than leaving them insecure. We consider that enforcement of liability is one of the most promising ways to drive this, as otherwise there is little incentive for the required commercial resources to be expended in an increasingly fiercely competitive marketplace. However in the absence of significantly improved practitioner education at all phases of the system life cycle, enforcement of liability will fail to deliver real improvement as the competence to support it will be absent. Therefore the capacity to deliver to the required standards must come first.

In summary, we consider that changes to education, economics and legislation are the necessary routes to success, but motivation at the highest level is the key to action. To the proposal by one of us at a secure development conference in the UK that all software development should abide by security principles and secure development should be the only approach taught, a government representative responded that it would be too expensive. We agree that to address security adequately is indeed going to be costly, and much of that cost is unavoidably likely to be borne by government. So everything depends on how important cyber security really is to you as a nation. As the old fairground saying goes “*you pays your money and you takes your choice.*” And you also accepts your consequences.

END OF RESPONSE

[52] http://www.bsigroup.com/en-GB/kitemark/product-testing/

[53] e.g. http://www.theregister.co.uk/2015/10/21/german_govt_mulls_security_tests_of_sohopeless_routers/