Introduction to CyBOK
Issue 1.0

Andrew Martin | University of Oxford

Awais Rashid | University of Bristol

Howard Chivers | University of York

George Danezis | University College London

Steve Schneider | University of Surrey

Emil Lupu | Imperial College London

COPYRIGHT

Cyber security is becoming an important element in curricula at all education levels. However, the foundational knowledge on which the field of cyber security is being developed is fragmented, and as a result, it can be difficult for both students and educators to map coherent paths of progression through the subject. By comparison, mature scientific disciplines like mathematics, physics, chemistry, and biology have established foundational knowledge and clear learning pathways. Within software engineering, the IEEE Software Engineering Body of Knowledge [1] codifies key foundational knowledge on which a range of educational programmes may be built. There are a number of previous and current efforts on establishing skills frameworks, key topic areas, and curricular guidelines for cyber security. However, a consensus has not been reached on what the diverse community of researchers, educators, and practitioners sees as established foundational knowledge in cyber security.

The Cyber Security Body of Knowledge (CyBOK) aims to codify the foundational and generally recognised knowledge on cyber security. In the same fashion as SWEBOK, CyBOK is meant to be a guide to the body of knowledge; the knowledge that it codifies already exists in literature such as textbooks, academic research articles, technical reports, white papers, and standards. Our focus is, therefore, on mapping established knowledge and not fully replicating everything that has ever been written on the subject. Educational programmes ranging from secondary and undergraduate education to postgraduate and continuing professional development programmes can then be developed on the basis of CyBOK.

This introduction sets out to place the 19 Knowledge Areas (KAs) of the CyBOK into a coherent overall framework. Each KA assumes a baseline agreement on the overall vocabulary, goals, and approaches to cyber security, and here we provide that common material which underpins the whole body of knowledege. We begin with an overview of cyber security as a topic, and some basic definitions, before introducing the knowledge areas. The KAs and their groupings into categories are, of course, not orthogonal and there are a number of dependencies across the KAs which are cross-referenced and also separately captured visually on the CyBOK web site (https://www.cybok.org). We then discuss how the knowledge in the KAs can be deployed to understand the means and objectives of cyber security, mitigate against failures and incidents, and manage risks.

Although we have necessarily divided the CyBOK into a number of discrete Knowledge Areas (KAs), it is clear that there are many inter-relationships among them. Those with professional responsibility for one area must typically have at least a moderate grasp of the adjacent topics; someone responsible for architecting a secure system must understand many. There are a number of *unifying principles* and crosscutting themes — *security economics; verification and formal methods; and security architecture and lifecycle* — that underpin the development of systems that satisfy particular security properties. We conclude the introduction by discussing such principles and themes.

1 CYBER SECURITY DEFINITION

The CyBOK Knowledge Areas assume a common vocabulary and core understanding of a number of topics central to the field. Whilst this *Body of Knowledge* is *descriptive* of existing knowledge (rather than seeking to innovate, or constrain), it is evident that use of widely-shared terminology in an established concept map is crucial to the development of the discipline as a whole. Since our main aim is to provide a *guide* to the Body of Knowledge, we will provide references to other definitions, rather than introducing our own.

Cyber security has become an encompassing term, as our working definition illustrates:

> **Definition:** ***Cyber security*** refers to the protection of information systems (hardware, software and associated infrastructure), the data on them, and the services they provide, from unauthorised access, harm or misuse. This includes harm caused intentionally by the operator of the system, or accidentally, as a result of failing to follow security procedures.
>
> *UK National Cyber Security Strategy [2]*

This is a succinct definition but expresses the breadth of coverage within the topic. Many other definitions are in use, and a document from ENISA [3] surveys a number of these.

The consideration of human behaviours is a crucial element of such a definition—but arguably still missing is a mention of the impact on them from loss of information or reduced safety, or of how security and privacy breaches impact trust in connected systems and infrastructures. Moreover, security must be balanced with other risks and requirements—from a human factors perspective there is a need not to disrupt the primary task.

A large contributor to the notion of cyber security is *Information Security*, widely regarded as comprised of three main elements:

> **Definition:** ***Information security.*** Preservation of confidentiality, integrity and availability of information.
>
> In addition, other properties, such as authenticity, accountability, non-repudiation, and reliability can also be involved.
>
> *ISO 27000 definition [4]*

For definitions of the subsidiary terms, the reader is referred to the ISO 27000 definitions [4].

Through the developing digital age other 'securities' have had prominence, including *Computer Security* and *Network Security*; related notions include *Information Assurance*, and *Systems Security* — perhaps within the context of *Systems Engineering* or *Security Engineering*. These terms are easily confused, and it seems that often one term is used when another is meant.

Many of those terms were subject to the criticism that they place an over-reliance on technical controls, and focus almost exclusively on *information*. Stretching them to relate to cyber-physical systems may be taking them too far: indeed, our working definition above privileges the notion of *information* (whilst also mentioning *services*) — whereas in the case of network-connected actuators, the pressing challenge is to prevent unwanted *physical actions*.

Moreover, in some accounts of the topic, cyberspace is best understood as a 'place' in which business is conducted, human communications take place, art is made and enjoyed, relationships are formed and developed, and so on. In this place, cyber crime, cyber terrorism, and cyber war may occur, having both 'real' and 'virtual' impacts. Taken as a whole, the CyBOK delineates a large range of topics which appear to be within the broad scope of *cyber security*, even if a succinct reduction of those into a short definition remains elusive. The full scope of CyBOK may serve as an extended definition of the topic—as summarised next.

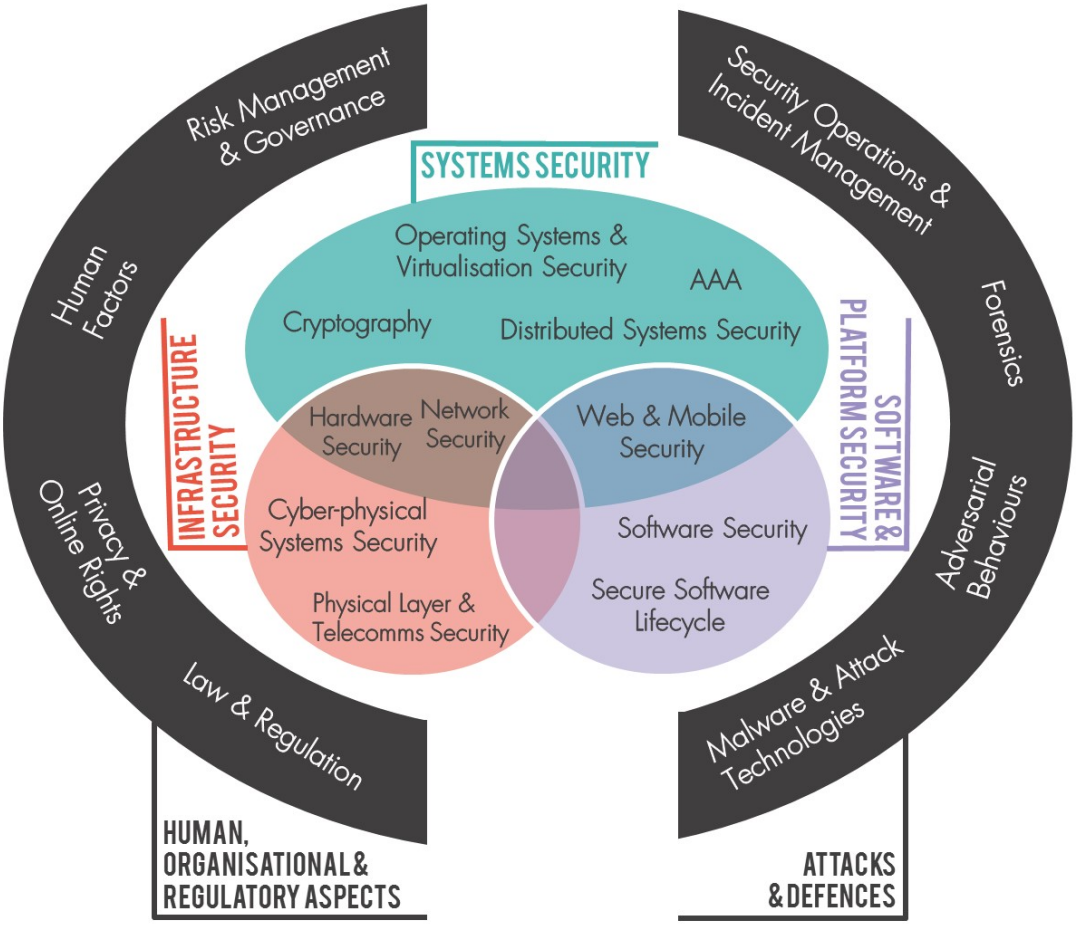

Figure 1: The 19 Knowledge Areas (KAs) in the CyBOK Scope

2 CYBOK KNOWLEDGE AREAS

The CyBOK is divided into nineteen top-level Knowledge Areas (KAs), grouped into five broad categories, as shown in Figure 1. Clearly, other possible categorisations of these KAs may be equally valid, and ultimately some of the structure is relatively arbitrary. The CyBOK Preface describes the process by which these KAs were identified and chosen.

Our categories are not entirely orthogonal. These are intended to capture knowledge relating to cyber security *per se*: in order to make sense of some of that knowledge, auxiliary and background knowledge is needed — whether in the design of hardware and software, or in diverse other fields, such as law.

	Human, Organisational, and Regulatory Aspects
Risk Management & Governance	Security management systems and organisational security controls, including standards, best practices, and approaches to risk assessment and mitigation.
Law & Regulation	International and national statutory and regulatory requirements, compliance obligations, and security ethics, including data protection and developing doctrines on cyber warfare.
Human Factors	Usable security, social & behavioural factors impacting security, security culture and awareness as well as the impact of security controls on user behaviours.
Privacy & Online Rights	Techniques for protecting personal information, including communications, applications, and inferences from databases and data processing. It also includes other systems supporting online rights touching on censorship and circumvention, covertness, electronic elections, and privacy in payment and identity systems.
	Attacks and Defences
Malware & Attack Technologies	Technical details of exploits and distributed malicious systems, together with associated discovery and analysis approaches.
Adversarial Behaviours	The motivations, behaviours, & methods used by attackers, including malware supply chains, attack vectors, and money transfers.
Security Operations & Incident Management	The configuration, operation and maintenance of secure systems including the detection of and response to security incidents and the collection and use of threat intelligence.
Forensics	The collection, analysis, & reporting of digital evidence in support of incidents or criminal events.
	Systems Security
Cryptography	Core primitives of cryptography as presently practised & emerging algorithms, techniques for analysis of these, and the protocols that use them.
Operating Systems & Virtualisation Security	Operating systems protection mechanisms, implementing secure abstraction of hardware, and sharing of resources, including isolation in multiuser systems, secure virtualisation, and security in database systems.
Distributed Systems Security	Security mechanisms relating to larger-scale coordinated distributed systems, including aspects of secure consensus, time, event systems, peer-to-peer systems, clouds, multitenant data centres, & distributed ledgers.
Authentication, Authorisation, & Accountability	All aspects of identity management and authentication technologies, and architectures and tools to support authorisation and accountability in both isolated and distributed systems.
	Software and Platform Security
Software Security	Known categories of programming errors resulting in security bugs, & techniques for avoiding these errors—both through coding practice and improved language design—and tools, techniques, and methods for detection of such errors in existing systems.
Web & Mobile Security	Issues related to web applications and services distributed across devices and frameworks, including the diverse programming paradigms and protection models.
Secure Software Lifecycle	The application of security software engineering techniques in the whole systems development lifecycle resulting in software that is secure by default.
	Infrastructure Security
Network Security	Security aspects of networking & telecommunication protocols, including the security of routing, network security elements, and specific cryptographic protocols used for network security.
Hardware Security	Security in the design, implementation, & deployment of general-purpose and specialist hardware, including trusted computing technologies and sources of randomness.
Cyber-Physical Systems Security	Security challenges in cyber-physical systems, such as the Internet of Things & industrial control systems, attacker models, safe-secure designs, and security of large-scale infrastructures.
Physical Layer & Telecommunications Security	Security concerns and limitations of the physical layer including aspects of radio frequency encodings and transmission techniques, unintended radiation, and interference.

Figure 2: Short descriptions of CyBOK Knowledge Areas

3 DEPLOYING CYBOK KNOWLEDGE TO ADDRESS SECURITY ISSUES

3.1 Means and objectives of cyber security

Implicit in the definitions above is that cyber security entails protection *against* an *adversary* or, possibly, against some other physical or random process. The latter implies some overlap between the notions of *safety* and *security*, although it is arguably possible to have either without the other. Within the security domain, if our modelling accounts for malice, it will necessarily encompass accidents and random processes. Therefore, core to any consideration of security is the modelling of these adversaries: their motives for attack, the threats they pose and the capabilities they may utilise.

In considering those threats, cyber security is often expressed in terms of instituting a number of *controls* affecting people, process, and technology. Some of these will focus on the *prevention* of bad outcomes, whereas others are better approached through *detection* and *reaction*. Selection of those controls is generally approached through a process of Risk Management (see below, and the Risk Management & Governance CyBOK Knowledge Area [5]) — although increasing emphasis is placed on Human Factors (see the Human Factors CyBOK Knowledge Area [6]), noting the need to leverage humans as a lynchpin for improving cyber security cultures, as well as supporting them to protect their privacy online (see the Privacy & Online Rights CyBOK Knowledge Area [7]).

Equally, security requires an analysis of *vulnerabilities* within the system under consideration: a (hypothetical) system without vulnerabilities would be impervious to all threats; a highly vulnerable system placed in totally benign circumstances (no threats) would have no security incidents, either.

The intended use of security controls gives rise to its own questions about whether they are deployed appropriately, and whether they are effective: these belong to the domain of *security assurance*, which has processes and controls of its own. These will involve residual risk analysis (see below, and the Risk Management & Governance CyBOK Knowledge Area [5]) which includes an attempt to measure and quantify the presence of vulnerabilities.

3.2 Failures and Incidents

When adversaries achieve their goal (wholly or partially) — when attacks succeed — the collection of security controls may be said to have failed. Alternatively, we may say that insufficient or ineffective controls were in place. Operationally speaking, one or more failures may give rise to a security incident. Typically such incidents may be described in terms of the harm to which they give rise: according to our definition of cyber security, these typically amount to harm from theft or damage of information, devices, services, or networks. The cyber-physical domain (see the Cyber-Physical Systems Security CyBOK Knowledge Area [8]) gives rise to many additional potential harms—harms to humans may come from either information, or from unintended physical action, or from both.

A significant sub-discipline of operational security considers detection of security failures, and reactions to them (remediation where possible). The Security Operations & Incident Management CyBOK Knowledge Area [9] addresses the context; the Malware & Attack Technology CyBOK Knowledge Area [10] deals with analysis of attack vectors while the Forensics CyBOK

Knowledge Area [11] considers the technical details and processes for post-attack analysis in a robust and reliable manner.

A recurrent theme in security analysis is that it is not sufficient to define good security controls solely within a particular abstraction or frame of reference: it is necessary also to consider what may happen if an adversary chooses to ignore that abstraction or frame.

This arises, for example, in communication *side channels*, where an adversary may infer much from capturing radio frequency emissions from a cable, say, without needing to tap that cable physically. Similar eavesdropping effects have been observed against cryptography implemented on smartcards: simple analysis of the power consumption of the processor as it addresses each bit in turn can be sufficient to disclose the cryptographic key (see Cryptography, Hardware Security and Software Security Knowledge Areas).

These problems occur at every level in the system design. In software, the *SQL injection attack* arises (see Software Security and Web & Mobile Security Knowledge Areas) because a string of characters intended to be interpreted as a database entry is forced to become a database command. Files holding secrets written by one application may give up those secrets when read by another, or by a general-purpose debugger or dump program.

Mathematical theories of refinement (and software development contracts) explore the relationship of an 'abstract' expression of an algorithm and a more 'concrete' version which is implemented: but security properties proven of the one may not be true of the other (for example, reducing uncertainty can increase information content and lead to the leak of information such as a cryptographic key), so great care must be taken in the construction of the theories. 'Black-box testing' relies on the same notion and, since it cannot possibly test every input, may easily miss the particular combination of circumstances which — by accident or design — destroys the security of the program.

Operational security of a system may be predicated upon the operators following a particular procedure or avoiding particular dangerous circumstances: there is an assumption that if people are told in a professional context (not) to do something, then they will (not) do it. This is demonstrably false (see the Human Factors CyBOK Knowledge Area [6]).

These — and an endless array of other — security problems arise because it is necessary to think (and design systems) using abstractions. Not only can no individual comprehend every detail of the operation of a networked computing system (from the device physics upwards), even if they had the requisite knowledge they must work in abstractions in order to make progress and avoid being overwhelmed with detail. But, for the majority of security controls, the abstraction is no more than a thinking tool: and so the adversary is able to disregard it entirely.

Since abstractions are usually built in layers (and computing systems are usually explicitly designed in that way), this is sometimes known as the 'layer below' problem [12] because the adversary often attacks the layer below the one in which the abstraction defining the control sits (see, for example, the threats and attacks discussed in the Operating Systems & Virtualisation CyBOK Knowledge Area [13] and the Hardware Security CyBOK Knowledge Area [14]).

3.3 Risk

There is no limit in principle to the amount of effort or money that might be expended on security controls. In order to balance these with the available resources and the harms and opportunities that might arise from emerging threats to security, a common over-arching approach to security analysis is a process of Risk Assessment — and selection of controls, a process of Risk Management. These are explored in depth in the Risk Management & Governance CyBOK Knowledge Area [5].

As with any process of risk management, a key calculation relates to expected impact, being calculated from some estimate of likelihood of events that may lead to impact, and an estimate of the impact arising from those events. The likelihood has two elements: the presence of *vulnerabilities* (known or unknown—the latter not always being capable of being mitigated), and the nature of the *threat*. The management response to the risk assessment may take many forms, including additional controls to reduce the impact or likelihood of a threat, accepting the risk, or transferring/sharing it with a third party (e.g., insurance), or in some cases deciding not to proceed because all of these outcomes are unacceptable.

Security management encompasses all the management and security actions necessary to maintain the security of a system during its lifetime. Important in this context, but outside of the scope of the CyBOK, are quality management practices. Such practices are long-established in industry, essentially requiring that all work follows documented processes, and that the processes provide metrics which are, in turn, reviewed and used to correct processes that are not fit for purpose ('nonconformities').

The analogy between quality management and security is not perfect because the threat environment is not static; however, the trend is for security management standards such as ISO/IEC 27001 to embody standard quality management processes which are then specialised for security. The primary specialisation is the periodic use of risk management (see the Risk Management & Governance CyBOK Knowledge Area [5]), which must also take account of the changing threat environment. It is necessary to supplement periodic risk management with continuous measures of the effectiveness of the security processes. For example, system patching and maintenance can be continuously reviewed via vulnerability scanning, logs relating to failed access attempts, user lock-outs or password resets can provide indicators of the usability of security features.

The functions within a security management system can be grouped into Physical, Personnel, Information Systems and Incident Management and are a mixture of standard IT system management functions and those that are specific to cyber security.

Physical security includes physical protection of the system, including access control, asset management and the handling and protection of data storage media. These aspects are outside the scope of the CyBOK.

Personnel security is concerned with a wide range of security usability and behaviour shaping, including education and training (see the Human Factors CyBOK Knowledge Area [6]). It also includes formal human-resource management elements such as the selection and vetting of staff, terms and conditions of acceptable usage for IT systems (see the Law & Regulation CyBOK Knowledge Area [15]) and disciplinary sanctions for security breaches.

Information system management includes access management (see the Authentication, Authorisation & Accountability (AAA) CyBOK Knowledge Area [16]) and system logging (see the Security Operations & Incident Management CyBOK Knowledge Area [9]). The audit function is

divided into security monitoring (see the Security Operations & Incident Management CyBOK Knowledge Area [9]) and other IT functions, such as volumetric review for system provisioning. Management of the information system also involves standard IT functions such as backup and recovery, and the management of supplier relationships.

Incident management functions (see the Security Operations & Incident Management CyBOK Knowledge Area [9]) are specific to cyber security and include security monitoring, incident detection and response.

4 PRINCIPLES

Sound thinking and good practice in security has been codified by a number of authors. The principles they describe touch many different KAs, and taken together help to develop a holistic approach to the design, development, and deployment of secure systems.

4.1 Saltzer and Schroeder Principles

The earliest collected design principles for engineering security controls were enumerated by Saltzer and Schroeder in 1975 [17]. These were proposed in the context of engineering secure multi-user operating systems supporting confidentiality properties for use in government and military organisations. This motivation does bias them in some ways, however they have also stood the test of time in being applicable to the design of security controls much more broadly.

The eight principles they enumerate are as follows:

- *Economy of mechanism.* The design of security controls should remain as simple as possible, to ensure high assurance. Simpler designs are easier to reason about formally or informally, to argue correctness. Further, simpler designs have simpler implementations that are easier to manually audit or verify for high assurance. This principle underlies the notion of Trusted Computing Base (TCB) — namely the collection of all software and hardware components on which a security mechanism or policy relies. It implies that the TCB of a system should remain small to ensure that it maintain the security properties expected.
- *Fail-safe defaults.* Security controls need to define and enable operations that can positively be identified as being in accordance with a security policy, and reject all others. In particular, Saltzer and Schroeder warn against mechanisms that determine access by attempting to identify and reject malicious behaviour. Malicious behaviour, as it is under the control of the adversary and will therefore adapt, is difficult to enumerate and identify exhaustively. As a result basing controls on exclusion of detected violation, rather than inclusion of known good behaviour, is error prone. It is notable that some modern security controls violate this principle including signature based anti-virus software and intrusion detection.
- *Complete mediation.* All operations on all objects in a system should be checked to ensure that they are in accordance with the security policy. Such checks would usually involve ensuring that the subject that initiated the operation is authorised to perform it, presuming a robust mechanism for authentication. However, modern security controls may not base checks on the identity of such a subject but other considerations, such as holding a ‘capability’.

- *Open design.* The security of the control must not rely on the secrecy of how it operates, but only on well specified secrets or passwords. This principle underpins cyber security as a field of open study: it allows scholars, engineers, auditors, and regulators to examine how security controls operate to ensure their correctness, or identify flaws, without undermining their security. The opposite approach, often called 'security by obscurity', is fragile as it restricts who may audit a security control, and is ineffective against insider threats or controls that can be reverse engineered.
- *Separation of privilege.* Security controls that rely on multiple subjects to authorise an operation, provide higher assurance than those relying on a single subject. This principle is embodied in traditional banking systems, and carries forward to cyber security controls. However, while it is usually the case that increasing the number of authorities involved in authorising an operation increases assurance around integrity properties, it usually also decreases assurance around availability properties. The principle also has limits, relating to over diluting responsibility leading to a 'tragedy of the security commons' in which no authority has incentives to invest in security assuming the others will.
- *Least privilege.* Subjects and the operations they perform in a system should be performed using the fewest possible privileges. For example, if an operation needs to only read some information, it should not also be granted the privileges to write or delete this information. Granting the minimum set of privileges ensures that, if the subject is corrupt or software incorrect, the damage they may do to the security properties of the system is diminished. Defining security architectures heavily relies on this principle, and consists of separating large systems into components, each with the least privileges possible — to ensure that partial compromises cannot affect, or have a minimal effect on, the overall security properties of a whole system.
- *Least common mechanism.* It is preferable to minimise sharing of resources and system mechanisms between different parties. This principle is heavily influenced by the context of engineering secure multi-user systems. In such systems common mechanisms (such as shared memory, disk, CPU, etc.) are vectors for potential leaks of confidential information from one user to the other, as well as potential interference from one user into the operations of another. Its extreme realisation sees systems that must not interfere with each other being 'air-gapped'. Yet, the principle has limits when it comes to using shared infrastructures (such as the Internet), or shared computing resources (such as multi-user operating systems, that naturally share CPUs and other resources).
- *Psychological acceptability.* The security control should be naturally usable so that users 'routinely and automatically' apply the protection. Saltzer and Schroeder, specifically state that 'to the extent that the user's mental image of his protection goals matches the mechanisms he must use, mistakes will be minimised'. This principle is the basis for the Human Factors CyBOK Knowledge Area [6].

Saltzer and Schroeder also provide two further principles, but warn that those are only imperfectly applicable to cyber security controls:

- *Work factor.* Good security controls require more resources to circumvent than those available to the adversary. In some cases, such as the cost of brute forcing a key, the work factor may be computed and designers can be assured that adversaries cannot be sufficiently endowed to try them all. For other controls, however, this work factor is harder to compute accurately. For example, it is hard to estimate the cost of a corrupt insider, or the cost of finding a bug in software.

- *Compromise recording.* It is sometimes suggested that reliable records or logs, that allow detection of a compromise, may be used instead of controls that prevent a compromise. Most systems do log security events, and security operations heavily rely on such reliable logs to detect intrusions. The relative merits — and costs — of the two approaches are highly context-dependent.

Those principles in turn draw on much older precedents such as Kerckhoff's principles relating to cryptographic systems [18]. Kerchoff highlights that cryptographic systems must be practically secure, without requiring the secrecy of how they operate (open design). He also highlights that keys should be short and memorable, the equipment must be easy to use, and applicable to telecommunications — all of which relate to the psychological acceptability of the designs.

4.2 NIST Principles

More contemporary principles in systems design are enumerated by NIST[19, Appendix F]. They incorporate and extend the principles from Saltzer and Schroeder. They are categorised into three broad families relating to: 'Security Architecture and Design' (i.e., organisation, structure and interfaces); 'Security Capability and Intrinsic Behaviours' (i.e., what the protections are about); and 'Life Cycle Security' (i.e., those related to process and management). As such those principles specifically refer to security architecture, specific controls, as well as engineering process management.

A number of the NIST principles map directly to those by Saltzer and Schroeder, such as Least Common Mechanism, Efficiently Mediated Access, Minimised Sharing, Minimised Security Elements, Reduced Complexity, Least Privilege, Secure Defaults and Predicate Permission, and Acceptable Security.

Notably, new principles deal with the increased complexity of modern computing systems and emphasise clean modular design, i.e. with Clear Abstraction, Modularity and Layering, Partially Ordered Dependencies, Secure Evolvability. Other principles recognise that not all components in a secure system may operate at the same level of assurance, and call for those to benefit from a Hierarchical Trust structure, in which the security failure of some components does not endanger all properties in the system. The principle of Inverse Modification Threshold states that those components that are the most critical to security, should also be the most protected against unauthorised modification or tampering. Hierarchical protection states that least critical security components need not be protected from more critical ones.

The NIST framework also recognises that modern systems are interconnected, and provides principles of how to secure them. These should be networked using Trusted Communication Channels. They should enjoy Secure Distributed Composition, meaning that if two systems that enforce the same policy are composed, their composition should also at least enforce the same policy. Finally, the principle of Self-Reliant Trustworthiness states that a secure system should remain secure even if disconnected from other remote components.

The NIST principles expand on what types of security mechanisms are acceptable for real-world systems. In particular the principles of Economic Security, Performance Security, Human Factored Security, and Acceptable Security state that security controls should not be overly expensive, overly degrade performance, or be unusable or otherwise unacceptable to users. This is a recognition that security controls support functional properties of systems and are not a goal in themselves.

Besides principles, NIST also outlines three key security architecture strategies. The Reference Monitor Concept is an abstract control that is sufficient to enforce the security properties of a system. Defence in Depth describes a security architecture composed on multiple overlapping controls. Isolation is a strategy by which different components are physically or logically separated to minimise interference or information leakage.

Both NIST, as well as Saltzer and Schroeder, highlight that principles provide guidance only, and need to be applied with skill to specific problems at hand to design secure architectures and controls. Deviation from a principle does not automatically lead to any problems, but such deviations need to be identified to ensure that any issues that may arise have been mitigated appropriately.

4.3 Latent Design Conditions

As more and more cyber-physical systems are connected to other systems and the Internet, the inherent complexity emerging from such large-scale connectivity and the safety critical nature of some of the cyber-physical systems means other principles also become highly relevant. One such principle is that of *Latent Design Conditions* from research in the safety-critical systems domain by James Reason [20]. In the context of cyber security, latent design conditions arise from past decisions about a system (or systems). They often remain hidden (or unconsidered) and only come to the fore when certain events or settings align — in the case of cyber-physical systems security vulnerabilities being exposed as they become connected to other systems or the Internet. Reason refers to this as the Swiss Cheese model where different holes in the slices align. These issues are discussed further in the Human Factors CyBOK Knowledge Area [6]. The key point to note is that we can no longer just consider information loss as a potential consequence of cyber security breaches — but must also consider safety implications. Furthermore, security by design is not always a possibility and, as legacy systems become connected to other networked environments, one must consider the latent (insecure) design conditions that may be manifested and how to mitigate their impact.

4.4 The Precautionary Principle

As the participatory data economy leads to a range of innovative products and services, there are also growing concerns about privacy and potential misuse of data as has been highlighted by recent cases of interference in democratic processes. As such the *Precautionary Principle* — reflecting on the potential harmful effect of design choices before technological innovations are put into large-scale deployment — also becomes relevant. Designers must consider the security and privacy implications of their choices from conception, through to modelling, implementation, maintenance, evolution and also decommissioning of large-scale connected systems and infrastructures on which society increasingly relies. *Function creep* as the system evolves over its lifetime and its impact on the society-at-large must also be considered [21].

5 CROSSCUTTING THEMES

A number of topics and themes recur across various KAs — implicitly or explicitly — and provide a context or unification of ideas across those KAs which cuts across the structure chosen for the CyBOK. In a different decomposition of the CyBOK they might have been KAs in their own right. These are an important part of the body of knowledge, and so we document here the most substantial of them.

5.1 Security Economics

Economics of information security is a synthesis between computer and social science. It combines microeconomic theory, and to a lesser extent game theory, with information security to gain an in-depth understanding of the trade-offs and misaligned incentives in the design and deployment of technical computer security policies and mechanisms [22, 23]. For example, Van Eeten and Bauer studied the incentives of legitimate market players — such as Internet Service Providers (ISPs) and software vendors — when confronted with malware[1] and how the actions driven by such incentives lead to optimal or sub-optimal security for the wider interconnected system. Attacker economics is gaining importance as well (for example, [24, 25, 26]). Attacker economics exposes cost-benefit analyses of attackers to exploit vulnerabilities in the security of the victim target, to subsequently formulate protective countermeasures for law-abiding entities [27]. Lastly, there is the economics of deviant security [28]. This subdiscipline of attacker economics focuses on understanding how cyber criminals apply, i.e., practice, security to defend their systems and operations against disruption from law enforcement (e.g., resilience mechanisms built into botnets [29] or anti-forensics techniques [30]).

Security economics is, therefore, of high relevance across the various attacks and countermeasures discussed within the different KAs within CyBOK. It also plays a key role in understanding the cost of security to legitimate users of the system and to the cybercriminals — the strength of such a socio-technical approach is its acknowledgement that security is very much a human problem, and the cost versus benefits trade-offs are key to increasing our understanding of the decisions of defenders and attackers to respectively secure their systems or optimise attacks [22].

5.2 Verification and Formal Methods

Human frailty means that flaws frequently arise in system design or coding, and these often give rise to security vulnerabilities. The Software Engineering discipline has expended much effort in attempting to minimise the introduction of such faults, and to aid their early detection when they arise.

At its most basic, verification and validation of software systems entails testing — for consistency, uniform/predicted behaviour, and conformance to specifications. By its nature, such testing can never be complete or exhaustive on any realistic system, and it will necessarily be poor at finding deliberate flaws or systemic design failures. Approaches to verification and modelling seek to reason about designs and implementations in order to prove mathematically that they have the required security properties.

Formal methods are approaches to modelling and verification based on the use of formal

[1] http://www.oecd.org/internet/ieconomy/40722462.pdf

languages, logic and mathematics to express system and software specifications, and to model designs of protocols and systems. For security modelling and verification the adversary model is also incorporated into the reasoning, so that designs can be verified with respect to their security requirements in the context of particular classes of threat. Rigorous proofs establish that no attack of a particular class is possible, establishing security of the design against particular kinds of adversary. There are two principal approaches to formal modelling: computational, and symbolic.

The computational modelling approach [31] is close to the real system: it is a formal methodology at a more fundamental mathematical level, where messages are bitstrings, cryptographic functions are defined as functions on bitstrings, system participants are generally interactive Turing machines, and security parameters give asymptotic metrics to this methodology: the length of keys, complexity of algorithms, or measure of probabilities, vary with the security parameter. The adversary is considered to be a probabilistic polynomial time Turing machine. Precise definitions of cryptographic functions can be captured and analysed within the model. Security requirements are expressed as properties on the model including the adversary, and a security property is generally considered to hold if the probability that it does not hold is negligible in the security parameter.

Formal modelling has been used within the field of security for some decades, across many of the KAs classified in CyBOK under Systems Security, Infrastructure Security, and Software & Platform Security. For example, in the area of access control, the Bell-LaPadula model [32] provides an abstract model of the rules determining whether a subject with a certain security clearance should have a particular kind of access to an object with a given security classification. The aim of this model is to prevent data declassification; later work generalized this to methods for preventing certain information flows. Other access control models have been proposed to achieve other properties, such as integrity (e.g., the Biba model [33], or the Clark-Wilson model [34]). Formal methods enable key security properties to be expressed and proven in the formal model. Non-interference properties have been formalised [35] in terms of executions using transition systems, and system descriptions with transition system semantics can be evaluated against such properties.

The symbolic modelling approach is more abstract than the computational approach, and has been applied in a variety of flavours to the modelling and analysis of security protocols — sequences of interactions between agents to achieve a security goal such as authentication or key-exchange. Logic-based approaches such as the BAN logic [36] provide a language for expressing requirements such as confidentiality and authentication, facts around the sending and receiving of protocol messages, and inference rules to enable reasoning about correctness. Language-based approaches such as Applied Pi (e.g., [37, 38, 39]) provide languages to describe protocols explicitly, and construct a model of all possible executions including adversarial steps, in order to reason about the guarantees that the protocol can provide. Security properties are expressed in terms of what must be true for every execution in the model, e.g., if Bob believes at the end of a protocol run that he shares a session key with Alice, then the adversary is not also in possession of that session key.

Although the foundations of formal approaches are mature, the challenge has been in making them practical. The application of formal approaches requires the careful management of intricate detail, which in practice requires tool support to enable mechanised verification and to check proofs. Tool support for the symbolic approach comes either from general purpose formal methods tools applied to security problems such as Isabelle/HOL [40], or FDR [41], or from tools tailored specifically to security such as Tamarin [42] or ProVerif [43]. These tools

typically take either a theorem-proving approach or else a model-checking approach where the state space is explored exhaustively.

Verification using the computational modelling approaches have been more mathematical in nature, though tools such as CryptoVerif [44] and EasyCrypt [45] have now been developed to support computational proofs. The symbolic and computational approaches may be used together: an attack in a symbolic model will typically give rise to an attack in the computational model, so it is valuable to carry out a symbolic analysis of a system first in order to check for and design out any identified attacks. Once a symbolic model is verified, then some additional work is needed to establish security in the computational model. This can either be carried out directly, or through the application of general techniques such as computational soundness [46] which give conditions for symbolic results to apply to the computational model.

These tools are now becoming strong enough to verify deployed protocols such as TLS1.3, which has been verified using a combination of both approaches [47], but they still require expert guidance. Further development of the tool support is an active research area.

5.3 Security Architecture and Lifecycle

The word 'architecture' is used at all levels of detail within a system; here we are concerned with the high-level design of a system from a security perspective, in particular how the primary security controls are motivated and positioned within the system. This, in turn, is bound up with an understanding of the *systems* lifecycle, from conception to decommissioning. Within this, the secure *software* lifecycle is crucial (the subject of the Secure Software Lifecycle Knowledge Area).

The fundamental design decision is how a system is compartmentalised — how users, data, and services are organised to ensure that the highest risk potential interactions are protected by the simplest and most self-contained security mechanisms (see Section 4). For example, a network may be divided into front-office/back-office compartments by a network router or firewall that permits no inward connections from the front to the back. Such a mechanism is simpler and more robust than one that uses access controls to separate the two functions in a shared network.

The first step is to review the proposed use of the system. The business processes to be supported should identify the interactions between the users, data or services in the system. Potential high risk interactions between users (see the Adversarial Behaviours CyBOK Knowledge Area [48] and data should then be identified with an outline risk assessment (see the Risk Management & Governance CyBOK Knowledge Area [5]) which will also need to take account of external requirements such as compliance (see the Law & Regulation CyBOK Knowledge Area [15]) and contractual obligations. If users with a legitimate need to access specific data items also pose a high risk to those items, or if any user has unconstrained authority to effect an undesired security outcome, the business process itself must be revised. Often such cases require a 'two person' rule, for example, counter-authorisation for payments.

The next step is to group users and data into broad categories using role-access requirements, together with formal data classification and user clearance. Such categories are potential system compartments, for example, Internet users and public data, or engineers and design data. Compartments should ensure that the highest risk user-data interactions cross compartment boundaries, and that common user-data interactions do not. Such compartments are usually enforced with network partitioning controls (see the Network Security CyBOK Knowledge Area [49]). Detailed design is then required within compartments, with the first steps being

to focus on concrete user roles, data design and access controls (see the Authentication, Authorisation & Accountability (AAA) CyBOK Knowledge Area [16]), with more detailed risk assessments being conducted as the design matures.

Systems benefit from a uniform approach to security infrastructure, for example, the management of keys and network protocols (see the Network Security CyBOK Knowledge Area [49]), resource management and coordination (see the Distributed Systems Security CyBOK Knowledge Area [50]), roles (see the Authentication, Authorisation & Accountability (AAA) CyBOK Knowledge Area [16]), user access (see the Human Factors CyBOK Knowledge Area [6]), and intrusion detection (see the Security Operations & Incident Management CyBOK Knowledge Area [9]). CyBOK provides important foundation knowledge in these areas, but neither this nor risk assessment are sufficient to motivate the detailed implementation of infrastructure; they need to be complemented by current good practice. In some industries best practice is mandated (e.g., the Payment Card Industries). In other cases it may be available from open sources (e.g., OWASP[2]) or as a result of corporate benchmarking.

Orthogonal to these concerns are a number of topics which relate to the context of the system development and operation. It is increasingly clear that a code of conduct, as prescribed by many professional bodies, offers a valuable framework for system designers and those who explore weaknesses and vulnerabilities within such systems. Initiatives around responsible research and innovation are gaining ground. The discovery of vulnerabilities necessitates a disclosure policy — and the parameters of responsible disclosure have prompted much debate, together with the role of this in a security equities process.

These broad consideration of architecture and lifecycle have been captured within the notions of 'security by design', and 'secure by default'[3]. The former term is often applied to detailed practices in software engineering, such as input checking, to avoid buffer overflows and the like (see the Secure Software Lifecycle CyBOK Knowledge Area [51]). More generally, consideration of security throughout the lifecycle, including in the default configuration 'out of the box' (although not much software is delivered in boxes these days), demonstrably leads to less insecurity in deployed systems.

> We invite the readers to read the detailed descriptions captured in the 19 Knowledge Areas that follow and utilise the methods, tools, techniques and approaches discussed therein when tackling the challenges of cyber security in the increasingly connected digital world that we inhabit.

[2] https://www.owasp.org
[3] A related notion is 'privacy by design' (see the Privacy & Online Rights CyBOK Knowledge Area [7]).

ACKNOWLEDGEMENTS.

The authors thank Erik van de Sandt for permission to use text from his PhD thesis [28] in the section on Security Economics.

REFERENCES

[1] P. Bourque, R. E. Fairley *et al.*, *Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0*. IEEE Computer Society Press, 2014.
[2] HM Government, UK, "National cyber security strategy 2016–2021," 2016. [Online]. Available: https://www.gov.uk/government/publications/national-cyber-security-strategy-2016-to-2021
[3] ETSI/CEN/CENELEC Cybersecurity Coordination Group (CSCG) and ENISA, "Definition of Cybersecurity gaps and overlaps in standardisation," ENISA, Report, Dec. 2015. [Online]. Available: https://www.enisa.europa.eu/publications/definition-of-cybersecurity
[4] ISO/IEC, "Information technology – security techniques – information security management systems – overview and vocabulary," ISO/IEC, – 27000:2018, 2018.
[5] P. Burnap, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Risk Management & Governance, version 1.0. [Online]. Available: https://www.cybok.org/
[6] M. A. Sasse, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Human Factors, version 1.0. [Online]. Available: https://www.cybok.org/
[7] C. Troncoso, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Privacy & Online Rights, version 1.0. [Online]. Available: https://www.cybok.org/
[8] A. Cardenas, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Cyber-Physical Systems Security, version 1.0. [Online]. Available: https://www.cybok.org/
[9] H. Debar, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Security Operations & Inicident Management, version 1.0. [Online]. Available: https://www.cybok.org/
[10] W. Lee, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Malware & Attack Technology, version 1.0. [Online]. Available: https://www.cybok.org/
[11] V. Roussev, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Forensics, version 1.0. [Online]. Available: https://www.cybok.org/
[12] D. Gollmann, *Computer Security*, 3rd ed. Wiley, 2011.
[13] H. Bos, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Operating Systems & Virtualisation, version 1.0. [Online]. Available: https://www.cybok.org/
[14] I. Verbauwhede, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Hardware Security, version 1.0. [Online]. Available: https://www.cybok.org/
[15] R. Carolina, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Law & Regulation, version 1.0. [Online]. Available: https://www.cybok.org/
[16] D. Gollmann, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Authentication, Authorisation & Accountability, version 1.0. [Online]. Available: https://www.cybok.org/
[17] J. H. Saltzer and M. D. Schroeder, "The protection of information in computer systems," *Proceedings of the IEEE*, vol. 63, no. 9, pp. 1278–1308, 1975. [Online]. Available: https://doi.org/10.1109/PROC.1975.9939
[18] A. Kerckhoffs, "La cryptographie militaire," *Journal des sciences militaires*, vol. 9, 1883.

[19] R. Ross, M. McEvilley, and J. C. Oren, "Systems security engineering: Considerations for a multidisciplinary approach in the engineering of trustworthy secure systems," NIST, Tech. Rep. NIST.SP.800-160 Volume 1, Nov. 2016. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-160v1
[20] J. Reason, *The human contribution: unsafe acts, accidents and heroic recoveries*. CRC Press, 2008.
[21] W. Pieters and A. van Cleeff, "The precautionary principle in a world of digital dependencies," *IEEE Computer*, vol. 42, no. 6, pp. 50–56, 2009.
[22] R. Anderson and T. Moore, "The economics of information security," *Science*, vol. 314, no. 5799, pp. 610—-613, 2006.
[23] R. Anderson, "Why information security is hard-an economic perspective," in *17th Annual Computer Security Applications Conference (ACSAC 2001), 11-14 December 2001, New Orleans, Louisiana, USA*, 2001, pp. 358–365.
[24] D. Y. Huang, H. Dharmdasani, S. Meiklejohn, V. Dave, C. Grier, D. McCoy, S. Savage, N. Weaver, A. C. Snoeren, and K. Levchenko, "Botcoin: Monetizing stolen cycles," in *21st Annual Network and Distributed System Security Symposium, NDSS 2014, San Diego, California, USA, February 23-26, 2014*, 2014.
[25] P. Pearce, V. Dave, C. Grier, K. Levchenko, S. Guha, D. McCoy, V. Paxson, S. Savage, and G. M. Voelker, "Characterizing large-scale click fraud in ZeroAccess," in *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2014, pp. 141–152.
[26] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage, "Spamalytics: an empirical analysis of spam marketing conversion," in *Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008*, 2008, pp. 3–14.
[27] C. Herley and D. A. F. Florêncio, "Nobody sells gold for the price of silver: Dishonesty, uncertainty and the underground economy," in *8th Annual Workshop on the Economics of Information Security, WEIS 2009, University College London, England, UK, June 24-25, 2009*, 2009.
[28] E. van de Sandt, "Deviant security: The technical computer security practices of cyber criminals," Ph.D. dissertation, Department of Computer Science, University of Bristol, UK, 2019.
[29] C. Rossow, D. Andriesse, T. Werner, B. Stone-Gross, D. Plohmann, C. J. Dietrich, and H. Bos, "SoK: P2PWNED-modeling and evaluating the resilience of peer-to-peer botnets," in *IEEE Symposium on Security and Privacy*. IEEE, 2013, pp. 97–111.
[30] H. Berghel, "Hiding data, forensics, and anti-forensics," *Commun. ACM*, vol. 50, no. 4, pp. 15–20, 2007.
[31] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984. [Online]. Available: https://doi.org/10.1016/0022-0000(84)90070-9
[32] D. E. Bell and L. J. LaPadula, "Secure computer systems: Mathematical foundations," The MITRE Corporation, Bedford MA, Tech. Rep. ESD-TR-73-278, Nov. 1973.
[33] K. J. Biba, "Integrity consideration for secure computer systems," The MITRE Corporation, Bedford, MA, Tech. Rep. ESD-TR-76-372, MTR-3153, April 1977.
[34] D. D. Clark and D. R. Wilson, "A comparison of commercial and military computer security policies," in *Proceedings of the 1987 IEEE Symposium on Security and Privacy, Oakland, California, USA, April 27-29, 1987*, 1987, pp. 184–195. [Online]. Available: https://doi.org/10.1109/SP.1987.10001
[35] J. A. Goguen and J. Meseguer, "Security policies and security models," in *1982 IEEE Symposium on Security and Privacy, Oakland, CA, USA, April 26-28, 1982*, 1982, pp. 11–20.
[36] M. Burrows, M. Abadi, and R. M. Needham, "A logic of authentication," in *Proceedings of*

the Twelfth ACM Symposium on Operating System Principles, SOSP 1989, The Wigwam, Litchfield Park, Arizona, USA, December 3-6, 1989, 1989, pp. 1–13. [Online]. Available: https://doi.org/10.1145/74850.74852

[37] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–207, 1983. [Online]. Available: https://doi.org/10.1109/TIT.1983.1056650

[38] P. Y. A. Ryan, S. A. Schneider, M. H. Goldsmith, G. Lowe, and A. W. Roscoe, *Modelling and analysis of security protocols*. Addison-Wesley-Longman, 2001.

[39] M. Abadi and C. Fournet, "Mobile values, new names, and secure communication," in *Conference Record of POPL 2001: The 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, London, UK, January 17-19, 2001*, 2001, pp. 104–115. [Online]. Available: https://doi.org/10.1145/360204.360213

[40] "Isabelle," https://isabelle.in.tum.de/, accessed: 2019-09-15.

[41] "FDR4 — the CSP refinement checker," https://www.cs.ox.ac.uk/projects/fdr/, accessed: 2019-09-15.

[42] "Tamarin prover," http://tamarin-prover.github.io/, accessed: 2019-09-15.

[43] "Proverif: Cryptographic protocol verifier in the formal model," https://prosecco.gforge.inria.fr/personal/bblanche/proverif/, accessed: 2019-09-15.

[44] "Cryptoverif: Cryptographic protocol verifier in the computational model," https://prosecco.gforge.inria.fr/personal/bblanche/cryptoverif/, accessed: 2019-10-15.

[45] "Easycrypt: Computer-aided cryptographic proofs," https://www.easycrypt.info/trac/, accessed: 2019-10-15.

[46] M. Abadi and P. Rogaway, "Reconciling two views of cryptography (the computational soundness of formal encryption)," *J. Cryptology*, vol. 20, no. 3, p. 395, 2007. [Online]. Available: https://doi.org/10.1007/s00145-007-0203-0

[47] K. Bhargavan, B. Blanchet, and N. Kobeissi, "Verified models and reference implementations for the TLS 1.3 standard candidate," in *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, 2017, pp. 483–502. [Online]. Available: https://doi.org/10.1109/SP.2017.26

[48] G. Stringhini, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Adversarial Behaviours, version 1.0. [Online]. Available: https://www.cybok.org/

[49] S. Jha, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Network Security, version 1.0. [Online]. Available: https://www.cybok.org/

[50] N. Suri, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Distributed Systems Security, version 1.0. [Online]. Available: https://www.cybok.org/

[51] L. Williams, *The Cyber Security Body of Knowledge*. University of Bristol, 2019, ch. Secure Software Lifecycle, version 1.0. [Online]. Available: https://www.cybok.org/