Trusted Platform Module Based Privacy in Public Cloud: Challenges and Future Perspective

Devki Nandan Jha[*,ψ], Graham Lenton[ψ], James Asker[ψ], David Blundell[ψ], and David Wallom[*]

***Abstract*—Public cloud providers offer ready-to-use, easily scalable servers on demand for a variety of applications. Storing and processing private and sensitive data in the cloud brings multiple security issues and indeed these concerns currently prevent many users from utilising cloud resources. Improving both security and trust for users is increasingly important for cloud providers. In this context, we first investigate the classes of security threats encountered by cloud applications. We then analyse various software and hardware-based solutions to handle these security challenges and provide the user with a chain of trust. Our analysis shows pure software-based solutions do not sufficiently mitigate the challenges of the cloud environment. Hardware-based solutions utilising the Trusted Platform Module (TPM) alleviate the issues however, it is challenging to implement in the public cloud environment. Finally, we introduce our TPM-SGX based approach that utilises software TPM and Software Guard Extension (SGX) to provide similar security as the hardware-based approach.**

***Index Terms*—Public Cloud, Privacy, Communication, Trusted execution, Trusted Platform Module, Software Guard Extension**

I. Introduction

Over the past 10 years, there has been a significant increase in the popularity of cloud services [1]. Enterprises increasingly desire the opportunities offered by cloud providers to provision core IT services without the requirement to consider capital purchase and with simplified setup and maintenance. Alongside these easily understood benefits though are a number of new challenges when considering cloud adoption policies. One of the most significant is the need to provide privacy and trust to the client application. This challenge is widespread enough that it is one of the leading impediments to the widespread uptake of cloud computing [2].

A component or resource in a cloud environment is considered to be secured if it maintains *a) confidentiality*, information is kept secret from unauthorised access; *b) integrity*, the state of resource/data will be in the same state as expected; and *c) availability*, the resource/data is available when required. A failure of any one of these properties renders the environment insecure. An external attacker can breach any of these properties and violate the system security.

A. Security Threats

To handle the system's privacy and security, it is important to understand the potential vulnerabilities a system may encounter. Although there are numerous security and privacy issues possible in the cloud environment, in this paper we focus on the 4 major classes, *1. network attack*, *2. malware attack*, *3. untrusted cloud administrator* and *4. data manipulation attack* as discussed below [3], [4].

1) Network attack: This is the most common form of attack in a public cloud environment as the client is accessing the services remotely [5]. An attacker can try to interrupt communication between the client and the cloud service provider by eavesdropping the communication channel or sniffing the data packets to gain unauthorised access of the private data (Man-in-the-middle attack). Using the confidential data, an attacker can also fool both the client and provider by replaying the previously used messages and performing malicious activities (Replay attack). The attacker can also steal the identity and later use that identity to pose as an authentic user (Identity theft).

Compromising the communication system and overloading the network with a large number of fake requests, also known as denial of service (DoS), is another common attack. DoS request exhausts network resources and makes the cloud services unavailable for authentic users.

2) Malware attack: In this attack, a small piece of software e.g. trojan horse or virus, designed to disrupt, damage, steal, or in general enforce some other malicious actions, gets inside the system. This malicious software triggers numerous data integrity issues affecting the cloud software stack. Although malware cannot normally penetrate the BIOS or hypervisor layer, it can cause serious harm to the operating system or any application executing on the operating system. It can also affect the run-time environment and obtain access to the secret keys which can later be used to perform various malicious activity. Certain worms can also propagate through the network thus affecting the subsequently communicated components (malicious coresident attack).

Integrity of a system can also be compromised unintentionally by misconfigured software, mishandled security patches and incompatible software updates which leads to inadvertent software behavior. Installation of a software from a hostile medium can also expose the system to various other type of attacks.

3) Untrusted cloud administrator: A possibly significant issue with executing applications and services in public cloud is the potential threat coming from inside the providers. It could be in the form of a malicious insider, an adversarial administrator or service misconduct. Since the physical resources are partitioned and scheduled to the virtual machines in a space/time shared manner, the cloud administrator handling the scheduling policies has full control of the virtual machine

[*]Author is with Oxford e-Research centre, University of Oxford, Oxford, UK. [ψ]Author is with 100 Percent IT Ltd., Newbury, UK.

resources. As the administrator possess privileged access to cloud services, malicious administrators can cause serious damage to the executing client applications or data. Allowing unauthorised privilege is another repercussion of an adversarial administrator. Here, an attacker can take advantage of the vulnerable cloud administrator to gain privileged access and later use this to initiate further attacks (root take-over attack). A list of recent cloud service providers mistakes causing security issues can be found in [6].

The hypervisor acts as the core of the cloud virtualisation and is accountable for maintaining isolation, handling CPU cycles, memory partitions and other services required for the management of the virtual machine. Compromise at the hypervisor level can thus endanger the security of all the virtual machines [7]. One class of attack is a Forking attack, where a snapshot of the system state is saved and forged with the current state; and a Rollback attack, where the system state is rolled back to the initial state; are common in cases when the hypervisor is compromised.

4) Data manipulation attack: In the current context where data is the main currency and the average cost to a business of a single Customer's personally identifiable information (PII) is circa $\$150$ [8], cloud applications need to have a strong data privacy policy avoiding any unrecognised infiltration. In addition to the users' application data and metadata, cloud environments also have system state and other privileged data hidden from users, it is necessary to keep all the data protected at the appropriate privilege level. Leakage of any of this data can be exploited by the attackers in many ways. For example manipulating the application's in-transit data may leads to execution failure in the application. Altering the metadata can also corrupt the whole virtual machine.

The above described attacks are not usually immediately apparent. The *Dwell Time*, or average time it takes for an attack to be noticed, is increasing. Currently this stands at 211 days with a further 67 days to contain it for a middle-staged cloud-based data breaches [8]. With Human error reported to be a factor in 95 % of data breaches and misconfigured cloud servers accounting for 19 % of breaches and the average cost of a breach standing at $\$4.8$ million for pubic cloud, it is clear that improving cloud security is a pressing concern.

To provide some standards for the user's data privacy and protection in cloud environment, International Organization for Standardization (ISO) defines a set of policies in the form of ISO 27017/ISO 27018 [9]. Most of the cloud service providers abide by the basic standards given in ISO 27017/ISO 27018 however, it is hard to detect any breaches as there is no auditing of human resource, supplier relationship or compliance activity. Any breach if occurs might be left undetected for a long time.

B. Existing Security Approach

To mitigate the risks caused by these attacks, an extensive range of defensive solutions are described in the literature [10], [11]. The most conventional way to handle these attacks is via software-based solutions e.g. antivirus software. These solutions have a database that contains a list of known malware or virus entities. The system is protected by scanning all the executable programs with the given database and isolating or removing the application with known threat entity. However, antivirus and black-listing solutions do not deal with zero-day vulnerabilities and attacks. In common with other software-based solutions they cannot also verify their own authenticity. Where a system is compromised by a malicious intruder, often their first action will be to disable these protections to hide the intruder's existence without the user's knowledge thus neutralising these so called protection mechanisms.

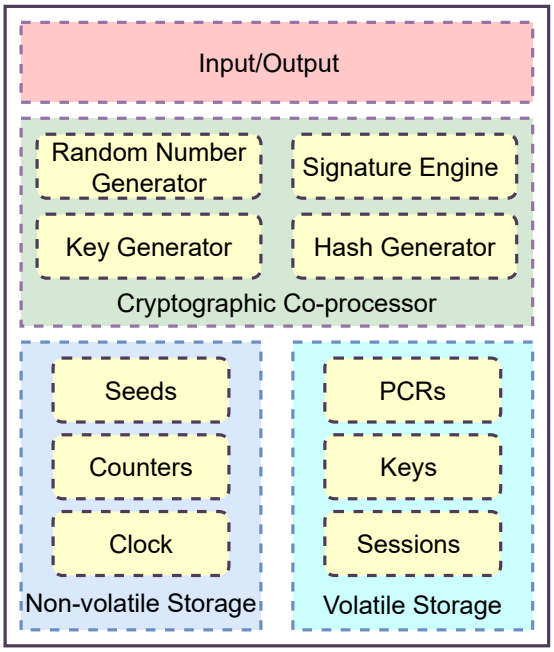

Fig. 1. Schematic diagram showing the internal architecture of a Trusted Platform Module.

In order to avoid the issues with the software-based defense mechanism, researchers are inclining towards an alternative hardware-based solution [12]. These approaches utilise system hardware, which are considered to be tamper-proof as the intruder has to physically interact with the hardware to implement the malicious activities, and are robust. A Trusted Computing Platform using a small hardware chip, *Trusted Platform Module (TPM)* is one of the best available solutions. Since the chip is integrated with the system, the security is bound to the hardware and is not easy to break. In the next section, we discuss about the details of TPM and how it tackles the above explained attacks.

II. Trusted Platform Module

The TPM is a cryptographic co-processor chip available for less than one *USD* which is included on almost every enterprise personal computer (PC), server and laptop motherboard. The specification for a TPM is provided by an industry consortium, Trusted Computing Group (TCG) [13]. A schematic diagram of the traditional TPM is given in Figure 1. It has 4 major components. The first component is

the *Input/Output* component which handles all the incoming requests and outgoing responses.

Cryptographic Co-processor is the major component which acts as a generator for a variety of elements including keys, hashes and signatures. There are two separate storage regions, *non-volatile* and *volatile* storage. *Non-volatile* storage stores the *Seeds*, *Counters* and *Clock* while the *volatile* storage contains *Platform Configuration Registers (PCRs)*, *Keys* and various *Session* information. The *Root of Trust* lies in the *Seed* and is used to create a set of keys using *Key Derivation Function (KDF)*. Since the *Seed* never leaves the TPM, it is considered to be the ultimate secret.

To handle the attacks given in Section I-A, the TPM performs the specific operations as given below:

1. *Key generation and storage* - One of the basic functions of the TPM is the generation of cryptographic keys. The keys here are generated using the *secret seed* or the *random number generator* with a key derivation function (KDF). The inclusion of entropy from either inside the TPM or via user input allows generation of an unlimited number of keys. User keys can also be imported to the TPM for further operation. These keys can be stored inside the TPM and used for various cryptographic operations. Due to the limited internal storage, the keys can also be removed from the memory and generated at run time for validation. Using key-based cryptography, most of the *network attacks* can be prevented.

2. *Integrity management* - To prevent the attack on the back-end of the system, it is necessary to measure and monitor the run-time system. The TPM can hash the system state and store the hash values in its PCR register. With every execution or at a defined interval, the hash is recomputed and compared with the previously stored hash. Since the PCR can not be rolled back to its previous state, any malicious activity can be easily detected. Performing the integrity management at the system boot time, ensures the secure system start-up thus assuring trust to the client.

3. *Remote attestation* - Although key-based cryptography can prevent various attacks, in many cases keys or the system administrators are compromised and the whole system placed at risk. It is important to prove the authenticity of an entity or key before granting access to the system. To prove the authenticity and trustworthiness of an entity or a key, the TPM performs an attestation. This process also uses integrity management in the form of a *TPM quote*: the hashed values of a number of PCRs which are signed by the TPM. If the TPM signature is validated as authentic, it is assured that the hash values are not altered. Since a nonce is also associated with this communication to guarantee freshness, any replay attack or system back-end delay can be easily identified.

4. *Authorisation of an entity* - Controlling the authorisation of an entity can prevent malicious intruders from altering the entity state and changing the data. Just using privilege-based authorisation will not protect against viruses embedded in an application. A TPM can mitigate these threats by controlling the access rights of an entity. A specific policy can be associated with the entity by setting a PCR to a particular value. The entity can only be accessible if the PCR is set to the desired value. Since multiple PCRs cannot be maliciously updated to the desired value, all the entities are protected from unauthorised access.

5. *User Identification and secure communication* - A TPM can also be used to identify the authenticity of a user. An identification key is associated with each user to prove their identity before establishing the communication. Since the identification key is derived from the TPM's trusted root key, any malicious user trying to invoke the system can be easily identified. Involvement of the cryptographic nonce prevents replay attacks. In this way, secure communication is established between a user and the cloud virtual machine.

III. Challenges with TPM-based privacy solutions in Public Cloud

With the rising popularity of virtual machines, The TCG has established a separate working group for the virtualised cloud environment [14]. However, there are numerous challenges for utilising TPM in a public cloud environment.

1. Unavailability of TPM - The main challenge is the unavailability of the TPM in public cloud environment as most of the public cloud providers do not operate hardware with physical TPMs.

2. Establishing Root of Trust - Even if the server has an in-built TPM which serves all the available virtual machines, it is complex to establish a root of trust for each of them. As the TPM chip is attached to the physical machine via a Low Pin Count (LPC) bus and the physical machine is virtualised to create numerous virtual machines, it is very complicated to distribute the TPM functionalities among all the associated virtual machines.

3. Processing time - TPM processing is quite slow in general. Virtualising the server's TPM for all the available virtual machines leads to huge delay which may not be acceptable.

An alternative option is to use a software-based TPM emulator. Numerous TPM emulators that follow the official TPM specifications are available to use. Although software-based TPMs are relatively fast compared to the physical TPM, they are not very secure. It can recognise any system state change but is still unable to protect its own keys. Since it is executing in the virtual machine as a process, any suitably-privileged user can retrieve or manipulate the keys stored by the software TPM. If the keys are accessed by a malicious intruder, the integrity of a software TPM and the overall system is at stake as the intruder can read the keys and leak their sensitive information. There is no analog concept that can be adapted to use a TPM-based solution for public clouds. Consequently, it requires special consideration when using on a cloud platform.

IV. Composite TPM-SGX based privacy approach

To utilise the advantages provided by the software TPM while maintaining the necessary security and hardware based root of trust, we propose a composite TPM-SGX based privacy

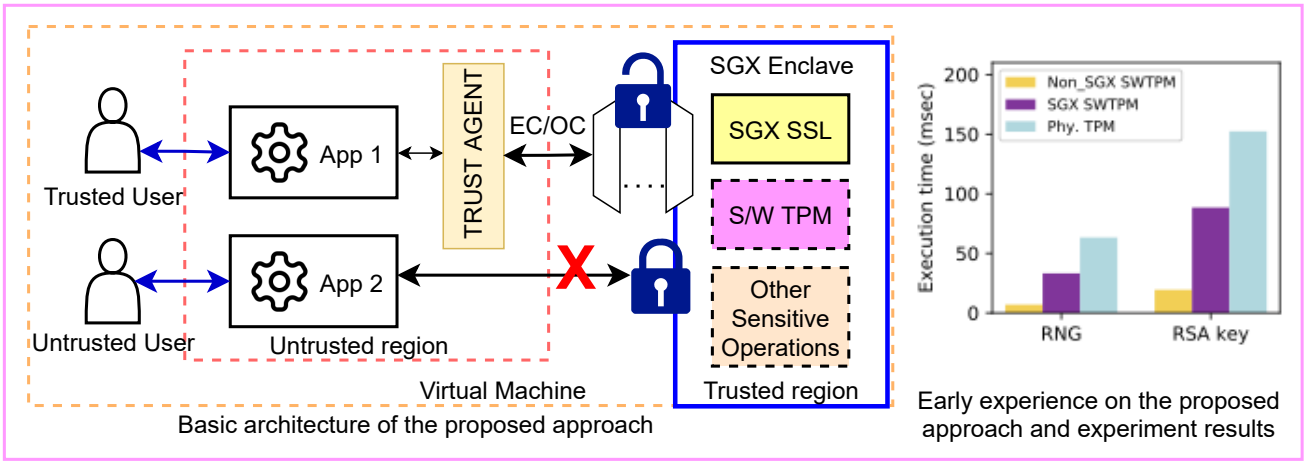

Fig. 2. Main functional elements and their interaction of our proposed framework. The Trust Agent establishes a communication channel with the SGX-based Software TPM using ECALLS (EC) and OCALLS (OC) allowing only trusted user to communicate. Initial results demonstrate that SGX-based software TPM is faster than the physical TPM allowing the same level of security.

framework. Software Guard Extensions (SGX) is another hardware-based privacy mechanism offered by the Trusted Execution Environment (TEE). SGX is a set of instructions added to some Intel processors that control memory access [15]. The SGX extension allow an application to instantiate a protected region of memory also known as the *Enclave*. The memory pages of an *Enclave* are non-addressable thus restricting access by any external application. Even the operating system can not access the protected *Enclave* memory as it is considered to be an external entity. The interaction with the SGX *Enclave* is carried out using special access permissions: *ECALLS* and *OCALLS*, which are defined at the initialisation time of *Enclave*.

A. SGX-based approach

After the first introduction of SGX in 2015, there was a surge in research on the application and improvement of SGX. Following the increasing popularity of SGX, many public cloud providers including Microsoft Azure, IBM cloud, and Alibaba cloud started providing SGX-based confidential computing services. Since the SGX *Enclave* provides a tamper-proof execution of any process while verifying it before execution, it acts as a suitable execution environment for a software TPM in the public cloud. As the SGX *Enclave* cannot be compromised even if the whole system including operating system is compromised, the integrity of hosted software TPM is always maintained.

Following this strategy, research has been proposed which combines the SGX execution with TPM. Hanon et al. [16] propose eTPM where software TPM is bound by SGX and physical TPM, however, this is not possible for public cloud as access to the physical TPM is not possible. Similar work is proposed by Juan et al. [17], where Secure virtual TPM (SvTPM) utilises SGX feature and provides a secure environment for QEMU and KVM. However, both of these approaches require handling and controlling the virtual machine monitor which is not feasible for public cloud environment.

B. Early Experience and Initial Results

Based on the existing trends, we propose a framework that utilises the features of software TPM and SGX and is also suitable for public cloud environment. A schematic diagram of our proposed approach is given in Figure 2. It consists of two main components *trusted* and *untrusted* following the SGX base. The software TPM is placed inside the trusted *Enclave* region to avoid tampering by any malicious entity. Placing the whole software TPM inside the SGX makes the framework generic enough to be used on any cloud environment (e.g. Azure confidential computing, AWS Nitro) without alteration. Another advantage is that this approach enables the full range of TPM functionality compared to a partial re-implementation. At the time of set up, the access rights of software TPM are restricted by enrolling it with a *Trust Agent*. Although the *Trust Agent* is executing in the untrusted region, the *Enclave* is constructed with the necessary permissions given to the *Trust Agent*. Also, our approach scales better than the shared physical TPM approach, as each virtual machine has its own SGX-TPM. During the enrolment process, the *Trust Agent* resets the software TPM and generates all the necessary root keys using the TPM in-built functionality. A certificate is also generated and provided to the *Trust Agent* which adds extra verification. Any further communication with the TPM is performed through the *Trust Agent*. The authenticity of the TPM is verified by SGX attestation before the launch process.

To show the effectiveness of our proposed model, we performed an initial experiment on Microsoft Azure confidential computing. The set up consists of a virtual machine with *Standard_DC2s* with 2 cores Intel(R) Xeon(R) E-2176G CPU @ 3.70GHz, 8 GB RAM and 60 GB Disk size with Ubuntu 18.04.5 LTS operating system. We employed Stefan Berger's Software TPM emulator, SWTPM [18]. We allow the SWTPM to be connected using only socket connections.

We performed the SWTPM experiment in two cases. In Case I, SWTPM is executing in the unprotected virtualised

environment while in Case II, it is executed inside the SGX trusted computing environment. In both cases, we performed two operations, the first one is to generate a random number of 1024 bytes and the second one is to create an RSA key of 1024 bytes from the storage hierarchy. The whole setup is repeated for 20 iterations and the result is averaged to get a mean value. The whole setup is also compared with the physical TPM case to get a better insight.

Figure 2 gives the initial result which shows that the SWTPM within SGX (Case II) is $5\times$ slower as compared to SWTPM alone (Case I). However, the execution time is faster by a factor of 1.5, which is a significant benefit for applications performing high-frequency TPM operations. Although, Software TPM is the fastest but does not protect its own keys and thus cannot guarantee privacy. Another experiment using *AES Finder*, a utility to find AES keys present in running process memory, shows that keys can be easily extracted for SWTPM while inaccessible for Case II where the keys are stored in SGX protected region.

V. Conclusion

This article has provided a comprehensive overview of the state of security and privacy in the public cloud environment. The work justifies the use of hardware-based security solutions as compared to software-based solutions. Although physical TPM-based solutions are considered to be the most suitable option, they are not appropriate for the public cloud environment. Integrating the software TPM inside SGX shows a greater potential as shown in the initial experience but requires more detailed study. In the future, we will evaluate the TPM-SGX approach on large scale in terms of scalability and flexibility. We will also explore various issues associated with SGX (e.g. side-channel attack) to guarantee trust in the public cloud solutions.

Acknowledgment

This research is supported by the Knowledge Transfer Partnership (KTP) project, *Trusted Public Cloud* (id 11289).

References

[1] "Cloud computing - statistics & facts," https://www.statista.com/topics/1695/cloud-computing/, accessed: 2021-03-02.

[2] J. A. De Guzman, K. Thilakarathna, and A. Seneviratne, "Security and privacy approaches in mixed reality: A literature survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 6, pp. 1–37, 2019.

[3] C. A. Ardagna, R. Asal, E. Damiani, and Q. H. Vu, "From security to assurance in the cloud: A survey," *ACM Computing Surveys*, vol. 48, no. 1, pp. 1–50, 2015.

[4] F. Zafar, A. Khan, S. U. R. Malik, M. Ahmed, A. Anjum, M. I. Khan, N. Javed, M. Alam, and F. Jamil, "A survey of cloud computing data integrity schemes: Design challenges, taxonomy and future trends," *Computers & Security*, vol. 65, pp. 29–49, 2017.

[5] J. B. Hong, A. Nhlabatsi, D. S. Kim, A. Hussein, N. Fetais, and K. M. Khan, "Systematic identification of threats in the cloud: A survey," *Computer Networks*, vol. 150, pp. 46–69, 2019.

[6] "Cloud service provider security mistakes," https://github.com/SummitRoute/csp_security_mistakes, accessed: 2022-01-10.

[7] D. Sgandurra and E. Lupu, "Evolution of attacks, threat models, and solutions for virtualized systems," *ACM Computing Surveys*, vol. 48, no. 3, pp. 1–38, 2016.

[8] "How much would a data breach cost your business?" https://www.ibm.com/uk-en/security/data-breach, accessed: 2021-10-22.

[9] T. Weil, "Taking compliance to the cloud—using iso standards (tools and techniques)," *IT Professional*, vol. 20, no. 6, pp. 20–30, 2018.

[10] P. R. Kumar, P. H. Raj, and P. Jelciana, "Exploring data security issues and solutions in cloud computing," *Procedia Computer Science*, vol. 125, pp. 691–697, 2018.

[11] A. Shamsoshoara, A. Korenda, F. Afghah, and S. Zeadally, "A survey on hardware-based security mechanisms for internet of things," *arXiv preprint arXiv:1907.12525*, 2019.

[12] L. Zhao and D. Lie, "Is hardware more secure than software?" *IEEE Security & Privacy*, vol. 18, no. 5, pp. 8–17, 2020.

[13] "Trusted computing group," https://trustedcomputinggroup.org/, accessed: 2021-09-21.

[14] "Trusted computing group, virtualized platform working group," https://trustedcomputinggroup.org/work-groups/virtualized-platform/, accessed: 2021-09-24.

[15] S. Chakrabarti, R. Leslie-Hurd, M. Vij, F. McKeen, C. Rozas, D. Caspi, I. Alexandrovich, and I. Anati, "Intel® software guard extensions (intel® sgx) architecture for oversubscription of secure memory in a virtualized environment," in *Proceedings of the Hardware and Architectural Support for Security and Privacy*, 2017, pp. 1–8.

[16] H. Sun, R. He, Y. Zhang, R. Wang, W. H. Ip, and K. L. Yung, "etpm: A trusted cloud platform enclave tpm scheme based on intel sgx technology," *Sensors*, vol. 18, no. 11, p. 3807, 2018.

[17] J. Wang, C. Fan, J. Wang, Y. Cheng, Y. Zhang, W. Zhang, P. Liu, and H. Hu, "Svtpm: A secure and efficient vtpm in the cloud," *arXiv preprint arXiv:1905.08493*, 2019.

[18] "Software tpm emulator," https://github.com/stefanberger/swtpm, accessed: 2021-03-08.

Biography

Devki Nandan Jha is a Postdoctoral research associate at the Oxford e-Research Centre, University of Oxford. He is also a Research Engineer at 100 Percent IT Ltd. He has PhD in Computer Science from School of Computing, Newcastle University, UK. His research interests include cloud computing, internet of things, big data analytics, security and privacy and machine learning. Contact him at Devki.Jha@eng.ox.ac.uk.

Graham Lenton is currently the Head of Engineering at CyberHive and 100 Percent IT Ltd. Before joining CyberHive, Graham led the development team at BBC Monitoring, UK. He has more than 25 years of Industry experience in development and management. His research interests include cloud computing, trusted computing, secure networking and software integration. Contact him at Graham.Lenton@cyberhive.com.

James Asker is a Software Developer at CyberHive and 100 Percent IT Ltd, and has 15 years of Industry experience in systems engineering and administration. His research interests are secure cloud computing and Linux kernel security. Contact him at James.Asker@100percentit.com.

David Blundell is founder, MD and CTO of CyberHive and 100 Percent IT Ltd. His current research interests are in secure distributed computing, post-quantum encryption and secure enclaves. Contact him at David.Blundell@cyberhive.com.

David Wallom is currently an Associate Director of Innovation with the Oxford e-Research Centre, University of Oxford. He was the Technical Director with the U.K. National Grid Service until the closure of the service, and is the Current Chair of the European Grid Infrastructure Federated Cloud Task Force. His current research interests include applications and reuse of e-infrastructure, as well as the application of high performance computing techniques and cybersecurity. Contact him at David.Wallom@oerc.ox.ac.uk.