applied sciences

Article

Virtualization Airborne Trusted General Computing Technology

Shuang Zhang [1,2], Yuanxun Wang [2], Xinyu Wan [2], Zhihui Li [2] and Yangming Guo [3,*]

[1] School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China
[2] Xi'an Aeronautical Computing Technique Research Institute, Xi'an 710068, China
[3] School of Cybersecurity, Northwestern Polytechnical University, Xi'an 710072, China
* Correspondence: yangming_g@nwpu.edu.cn

Abstract: Aircraft information service systems, such as airborne information systems, airborne integrated maintenance management systems, and cabin management systems, have greatly improved the ease of use and maintenance of civil aircraft. The current computing platforms used for accommodating these systems are unable to satisfy the multifaceted requirements of future information-based aircraft, such as energy conservation, emission reduction, high-performance computing, and information security protection, due to their high computing capacity, weight, and power consumption. Based on multi-core multi-threaded processors, a security hardware unit with microkernel virtualization technology and a virtualization airborne trusted general computing service architecture is proposed, and key technologies, including a high-performance processing and high-security hardware unit, virtualization management software unit, and virtualization security protection architecture were designed. By building a verification environment, the proposed platform was verified in terms of its application accommodation function, platform performance, and network security protection, for comparison with the existing platforms. The results showed that our method can fulfill the requirements of these technical indicators and is applicable, not only to new-generation civil aircraft, but also to unmanned aerial vehicles (UAVs) and emergency rescue aircraft with high-performance safety-critical computing needs.

Keywords: airborne general computing service (AGC); microkernel virtualization; virtualization security protection (VSP); trusted computing

check for updates

Citation: Zhang, S.; Wang, Y.; Wan, X.; Li, Z.; Guo, Y. Virtualization Airborne Trusted General Computing Technology. *Appl. Sci.* **2023**, *13*, 1342. https://doi.org/10.3390/app13031342

Academic Editors: Eui-Nam Huh and Teh-Lu Liao

Received: 26 November 2022
Revised: 12 January 2023
Accepted: 18 January 2023
Published: 19 January 2023

1. Introduction

With the evolution of the concept of information-based civil aircraft and the development of integrated avionics technology, information service systems with a safety level of D/E have been deployed in large civil aircraft [1,2], such as airborne information systems, airborne integrated maintenance management systems, and cabin management systems. Moreover, an airborne general computing service platform (AGCSP) offering a host environment for the above systems has been deployed, which consists in airborne general computing service modules (AGCSMs) [3]. As a line replaceable module (LRM), the AGCSM adopts a 32 bit single-threaded multi-core processor with an embedded Linux operating system, which does not support virtualized partitioning and disallows isolation among computing, storage, and network communication resources for applications. The existing first-generation AGCSP achieves security and isolation of the multiple airborne system applications by adding physical devices, where multiple AGCSMs are integrated in one platform, resulting in an excessively high platform weight and power consumption. In the future, the new generation of information-based civil aircraft need to meet green aviation goals, further lowering energy consumption, reducing carbon emissions, and cutting the weight and power consumption of the AGCSP. As the application scenarios increase, the information interactions among airborne information, airborne maintenance, cabin management, and airline operation service systems of the new-generation civil aircraft are becoming increasingly complex. Hence, stronger computing storage and network communication capabilities are required for AGCSP. According to the RTCA DO-326A specification [4] and RTCA Do-356A specification [5], new-generation civil aircraft should meet the requirements of information security airworthiness, which, as an AGCSP responsible for the coordination of air–ground business systems, need to have a network security protection capability. Unable to meet the demands of the new-generation civil aircraft, the existing AGCSP faces new challenges.

Owing to its high performance and low power consumption, the multi-core computing architecture has become the mainstream architecture of processors such as the PowerPC, X86, and ARM [6–8]. Virtualization technology can provide multiple applications with an execution environment featuring environmental isolation, scalable resource utilization, controllable operation [9,10], and platform portability on a single physical hardware platform. It hosts multiple guest operating systems and applications, to more effectively organize and utilize multi-core computing resources [11,12]. Multi-core processor-based virtualization has been gradually applied in safety critical systems [13–15]. As the data and service node in an aircraft network, the application of virtualization technology and a multi-core multi-threaded processor enables simultaneous operation of multiple virtual machines (VMs), operating systems, and applications using a single AGCSM. Embedded applications need to run in a trusted computing environment [16]. Trusted computing can achieve active immunity against attacks [17]. Based on the hardware security mechanism in the chip, it can actively detect and resist attacks, and provide data integrity and confidentiality assurance [18]. In the scenarios of the Internet of Things and edge cloud computing, trusted computing can provide an infrastructure for implementing a zero-trust architecture.

In reference [19], the authors focus on computing and network virtualization techniques for edge computing and delineate the primary factors for the selection of virtualization types in IoT frameworks. Virtualization techniques can provide temporal isolation, spatial isolation, and fault isolation. In reference [20], the authors compare dozens of major virtualization solutions for mixed-criticality systems from four categories: separation kernel and microkernel, general purpose, ARM TrustZone assisted, and lightweight virtualization. In the cloud and edge computing environments, the main threats include network traffic attacks, malware attacks, virtual network attacks, and DOS and DDOS attacks. Virtual machines mainly adopt virtual machine intrusion, hypervisor intrusion, and hybrid and other protection methods [21]. In [22], the authors present a detailed survey of the topics and challenges pertaining to security in hardware-assisted virtualization. They research risk modeling and threat evaluation and discuss the possible countermeasures and open challenges that remain in virtualization technologies. In [23], the authors analyze the different vulnerabilities that may affect the components of different virtualization models and identify related attacks. They propose several counter-measures and recommendations for hypervisor and virtual machine design. Reference [24] presents an approach for the experimental assessment of isolation properties against timing covert channels in the context of VxWorks MILS. In addition, the article presents an experimental evaluation, to show that timing covert channels are indeed feasible. Reference [25] proposes the S2H scheme, implements an efficient shared memory access, and proposes a "batch-grained" scheduling strategy, to ensure network performance in multi-tenant scenarios. However, the S2H scheme requires a lot of computing resources. Reference [26] proposed a zero-trust distributed engine control (ZT-DEC) strategy and architecture, which can mitigate against the risks of cyber-attacks and intrusions in safety critical systems. However, the ZTDEC architecture has limited performance and application scenarios. Reference [27] defined a safety and security co-engineering methodology for a systematic safety–security analysis of mixed-criticality systems running on heterogeneous high-performance embedded computing devices. However, the implementation of this methodology is complex and depends on the software stack. In [28], the authors used a concrete implementation to evaluate a multicore mixed-criticality system solution and compared it to a standard time-and-space-partitioning solution.

Hence, in response to the demand of new-generation information-based civil aircraft for airborne general computing services, a new platform with lower energy conservation, higher performance, and more secure protection is required. The major contributions of this paper are listed as follows:

(1) Establishing a virtualization airborne trusted general computing service architecture based on virtualization and trusted computing technologies.
(2) Designing key components, including embedded trusted multi-core computing hardware, embedded virtualization management software, virtualized trusted computing middleware, and embedded virtualized security access components.

The paper is structured as follows. Section 2 proposes the architecture of a virtualization airborne trusted general computing service platform. Section 3 describes the implementation methods for the key technologies. Section 4 presents the experimental results and a comparison to the AGCSP. Section 5 concludes the paper.

2. Architecture of the Virtualization Airborne Trusted General Computing Service Platform

To meet the application requirements of new-generation civil aircraft, the airborne general computing service should fulfill the following major technical indicators:

(1) High-performance computing capability and mass storage capacity.
(2) Provide an ARINC664P7 interface and gigabit Ethernet interface.
(3) Provide a secure and trusted computing environment with network traffic access control, attack detection, and defense capabilities.
(4) Support isolated resident operation environments for applications with different security levels.
(5) Over a one-fold improvement in computing performance compared to existing devices with an identical weight and power consumption.

Thus, clearly, the technical indicators of airborne general computing services, not only require an improvement in computing power, but should also meet isolation and security protection requirements. To this end, a high-performance multi-core processor is utilized to improve computing power, virtualization technology is employed to achieve isolation requirement, and trusted computing and virtualized security gateway technologies are adopted to meet security protection requirements, thereby designing a virtualization airborne trusted general computing service module (VATGCSM), as shown in Figure 1.

The VATGCSM architecture mainly encompasses a hardware facility layer, virtualization management layer, and VM operation layer. The hardware facility layer consists of a high-performance processing hardware unit and a high-security hardware unit. The virtualization management layer comprises a microkernel VM management unit and a virtualization layer security middleware unit. The VM operation layer can execute multiple VMs, including the virtualized security gateway VM and application hosting VM, of which the latter consists of an application unit, a guest operating system kernel unit, and a VM security middleware unit.

As the physical platform for the entire computing environment, the hardware facility layer offers the hardware resources necessary for operating the whole virtual environment, the hardware interfaces required by relevant businesses, as well as the functions of trusted computing modules. The virtualization management layer is the core scheduling layer in the virtualization mechanism, which provides the VM with functions such as hardware resource allocation, resource management, and communication security protection. The VM operation layer offers a resident environment for applications, which achieves separate residence for airborne application software with different security levels in the VMs.

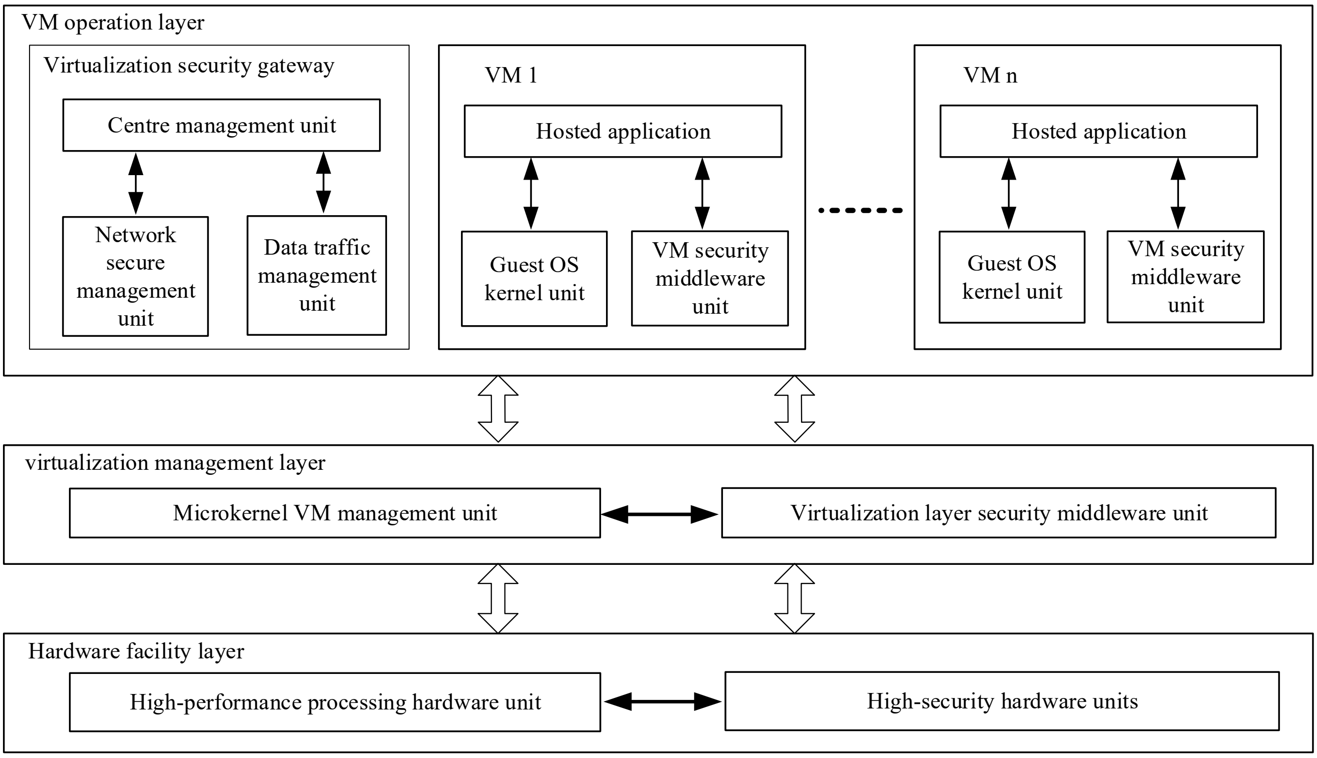

VM operation layer
Virtualization security gateway
Centre management unit
Network secure management unit
Data traffic management unit
VM 1
Hosted application
Guest OS kernel unit
VM security middleware unit
VM n
Hosted application
Guest OS kernel unit
VM security middleware unit
virtualization management layer
Microkernel VM management unit
Virtualization layer security middleware unit
Hardware facility layer
High-performance processing hardware unit
High-security hardware units

Figure 1. VAGCSP architecture.

The hardware facility and virtualization management layers are connected through an internal logical interface. The high-performance processing hardware unit provides high-performance processing capability to the microkernel VM management unit, which also offers trusted computing services for the virtualization layer security middleware unit, by communicating with the high-security hardware unit.

The virtualization management layer interacts with the VM operation layer through the system call interface. On the one hand, the virtualization management layer provides the VMs in the VM operation layer with allocation and management functions for virtualized hardware resources via the API interface, which is predefined by the microkernel VM management unit, and provides running environment support for the VM operation. On the other hand, it offers security protection for the communication among different VMs in the VM operation layer via the system call interface of the virtualization layer security middleware unit.

3. Key Technologies

3.1. High-Performance Processing and High-Security Hardware Units

The high-performance processing hardware unit and the high-security hardware unit constitute the hardware facility layer, of which the former offers basic hardware support for the operation of the entire virtual environment and the latter offers security support for the entire virtual environment. Figure 2 displays the relevant architecture.

The high-performance processing hardware unit comprises an ×86 architecture-based embedded multi-core processing module and gigabit Ethernet interfaces. The embedded multi-core processing module provides a high-performance processing capacity, with its four-core eight-thread processor and 16 GB memory, providing a mass storage capacity with 4 TB SSD, etc. Integrated with a 1 channel 100 Mbps ARINC664P7 interface and six-channel gigabit Ethernet interface, it also provides the resident airborne system applications with network communication services. Compared with AGCSP’s 950 g weight and 45 W power consumption, the VAGCSP weighs 930 g and consumes 42 W.

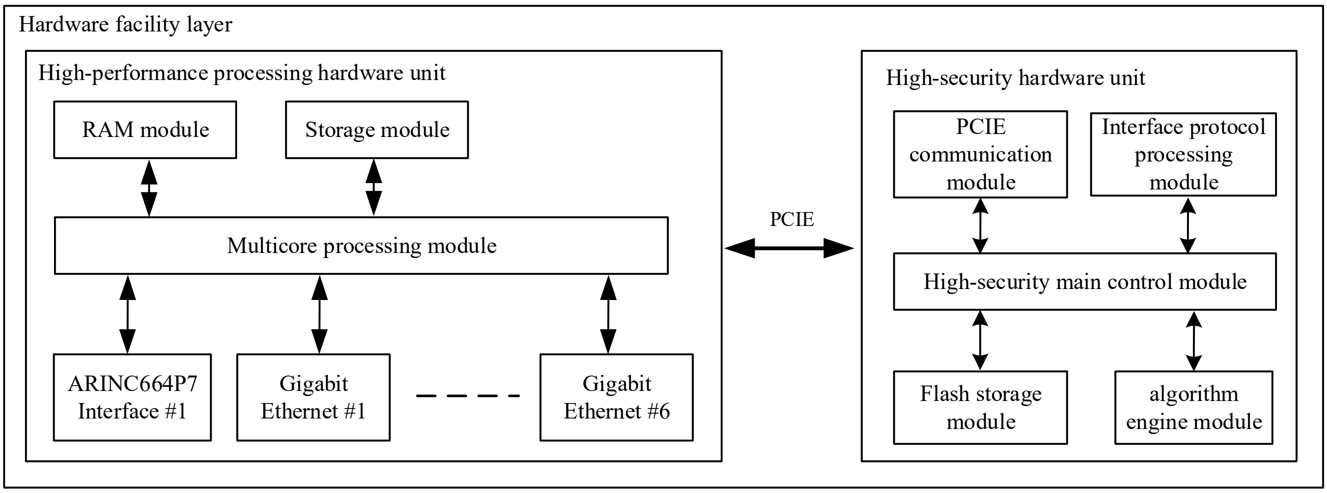

Hardware facility layer
High-performance processing hardware unit
RAM module
Storage module
Multicore processing module
ARINC664P7 Interface #1
Gigabit Ethernet #1
Gigabit Ethernet #6
PCIE
High-security hardware unit
PCIE communication module
Interface protocol processing module
High-security main control module
Flash storage module
algorithm engine module

Figure 2. Architecture of high-performance processing and high-security hardware units.

The high-security hardware unit is implemented using an independent hardware card, including the high-security main control module, PCIE communication module, interface protocol processing module, flash storage module, and algorithm engine module, which conforms to trusted computing standards. The major functions of the various parts are as follows:

(1) High-security main control module: implements the main control functions; calls the PCIE communication module to communicate with the embedded multi-core processing module; calls the interface protocol processing module to parse or encapsulate data packets, according to existing protocols; calls the flash storage module to store or read operation results; calls the algorithm engine module to request for high-security calculation and returns the corresponding results.

(2) PCIE communication module: connects high-performance processing hardware unit via the PCIE interface hardware; completes data communication with the embedded multi-core processing module; realizes data interaction by calling the interface and high-security main control module.

(3) Interface protocol processing module: implements established protocols; completes the encapsulation and parsing of data packets.

(4) Flash storage module: implements the flash storage access interface; realizes the storage of private data, such as certificates, digital signatures, and hashes.

(5) Algorithm engine module: achieves high-performance computing of cryptographic algorithms, such as hash, grouping, and public key algorithms; supports the international data encryption algorithm.

3.2. Virtualization Management Software Unit

Depending on the type of virtual machine manager (VMM), the mainstream virtualization architecture types can be classified into full virtualization and paravirtualization. In full virtualization mode, the existing operating system can be directly run without modifying the resident operating system. Regarding its disadvantages, VMM needs to consume a certain amount of computing resources when the guest operating system does not know that it is running in a virtualization environment. Paravirtualization mode requires partial modification of the operating system, so that the system kernel can avoid some instructions that are difficult to virtualize, thereby effectively improving the performance of the entire system. Its disadvantage is that modification of the operating system source code is necessary.

Considering that the ×86 processors have a specially optimized instruction set for controlling virtual process, they can support the hardware-assisted virtualization technology, so that the performance of full virtualization is improved to some extent. Thus, full virtualization design is adopted by the virtualization management software, and VM management is implemented by the microkernel VM management unit of the virtualization management layer. The VM management software unit uses the static configuration to bind the processor core, memory, and I/O devices. Through this design, the separate use of computing resources is realized and the distribution of computing resources between VMs is solved.

The microkernel VM management software unit comprises a memory management module, VM management module, I/O management module, and processor architecture module. The functions of the various parts are as follows:

(1) VM management module: implements functions such as VM management, allocation and management of memory, I/O, processors and other hardware resources, and VM security encapsulation.
(2) Processor architecture module: realizes processor virtualization; completes the translation of the processor instructions from the kernel unit of guest operating system and from the hardware facility layer; provides virtualized processing capabilities to VMs.
(3) Memory management module: implements memory virtualization; provides virtualized memory resources to VMS. Memory virtualization is primarily implemented using extent page table (EPT) technology, where hardware automatically maps the virtual addresses within VMs to physical addresses and the physical addresses of VMs to physical addresses of hosts using the VM page tables. In this way, the virtual addresses of VMs are translated into the physical addresses of hosts.
(4) I/O management module: implements virtual management of display devices, keyboards, mouse, gigabit Ethernet interfaces, USB devices, and other I/O devices in the hardware layer; provides virtualized device resources to VMs. Considering the usage frequency of I/O devices, software is used in the VMM to achieve their simulation, thereby providing VMs with support for these devices.

3.3. Virtualization Security Protection Architecture

To address security threats such as malicious network attacks and unauthorized network access, a virtualization security multilevel protection architecture was designed based on the high-security hardware unit, as shown in Figure 3, which encompasses the virtualization layer security middleware unit of virtualization management layer, the virtualization security gateway of VM layer, and the VM security middleware unit within VMs.

(1) The virtualization layer security middleware unit consists of the VM traffic management module, the VM security loading module, and the interface protocol processing module. The VM traffic management module can acquire all traffic between VMs, between VMs and hosts, as well as from outside hosts. It diverts all the acquired traffic to the virtualization security gateway for traffic processing, and then forwards the gateway-processed traffic. The VM security loading module implements the secure and reliable loading of VMs through trusted measurement of the eigenvalues of the guest operating system kernel unit in the VMs. By communicating with the high-security hardware unit in the hardware facility layer, the interface protocol processing module of high-security unit offers trusted computing support for the VM security loading module.
(2) The virtualization security gateway runs on the VM operation layer in the form of VMs, which implements security protection such as VM access control, intrusion prevention, and malicious code detection by processing the traffic acquired by the VM traffic management module.
(3) The VM security middleware unit, which runs in the kernel layer of the VM guest operating system, provides the resident applications with an inter-VM security communication service through the system call interface. When multiple VMs are started, the VM management software unit starts loading in sequence, according to the ID number of the VM. During the loading process, first the VM security loading module

is used to decrypt the image binary file of the VM, and then trusted measurement of the decrypted image file is implemented. When the trusted measurement of the VM passed, the VM image is loaded.

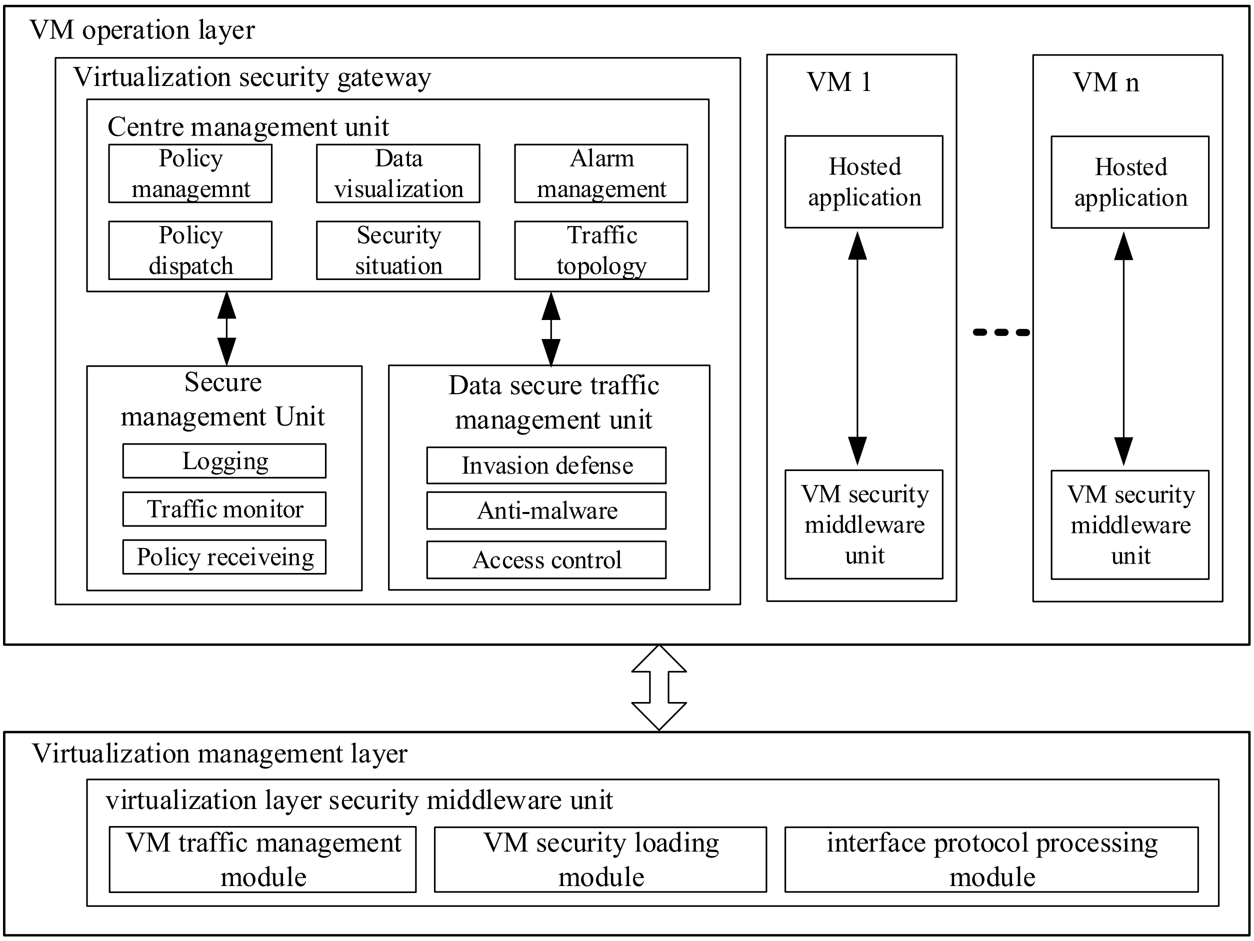

VM operation layer
Virtualization security gateway
Centre management unit
Policy managemnt
Data visualization
Alarm management
Policy dispatch
Security situation
Traffic topology
Secure management Unit
Logging
Traffic monitor
Policy receiveing
Data secure traffic management unit
Invasion defense
Anti-malware
Access control
VM 1
Hosted application
VM security middleware unit
VM n
Hosted application
VM security middleware unit
Virtualization management layer
virtualization layer security middleware unit
VM traffic management module
VM security loading module
interface protocol processing module

Figure 3. Virtualization security protection architecture.

4. Experimental Results and Analysis

4.1. Application Hosting Function Test

During the verification of application hosting function, multiple airborne applications with different safety levels are selected, such as the aircraft parameter distribution and file transfer service application of the airborne information system, the aircraft parameter acquisition application of airborne maintenance system, and the cabin parameter distribution and cabin lighting control applications of the cabin management system, to reside in multiple VMs of the VAGCSP, thereby verifying the VAGCSP's ability to host avionics system applications with low security levels. Figure 4 displays the verification environment.

The aircraft parameter distribution application resides in VM 1, which uses Ubuntu18.04 as the guest operating system. The aircraft parameter acquisition application resides in VM 2, with CentOS7 as the guest operating system. The cabin parameter distribution and lighting control applications reside in VM 3, whose guest operating system is Ubuntu18.04. The file transfer service application resides in VM 4, with Ubuntu18.04 as the guest operating system. To achieve isolation, each VM is configured with exclusive computing resources, including the processor cores, memory, and Ethernet interface. Table 1 lists the test items of the application hosting function. Adopting automated testing, the test case code developed based on Python was run on the testing equipment, and the application interaction data received during the test process were compared with the expected results to determine the functional correctness. Taking the test items for an aircraft parameter distribution application airborne information system as an example, the functional test results displayed in the testing equipment are shown in Figure 5. The outcomes of the VAGCSP's application hosting function test were consistent with the expected results.

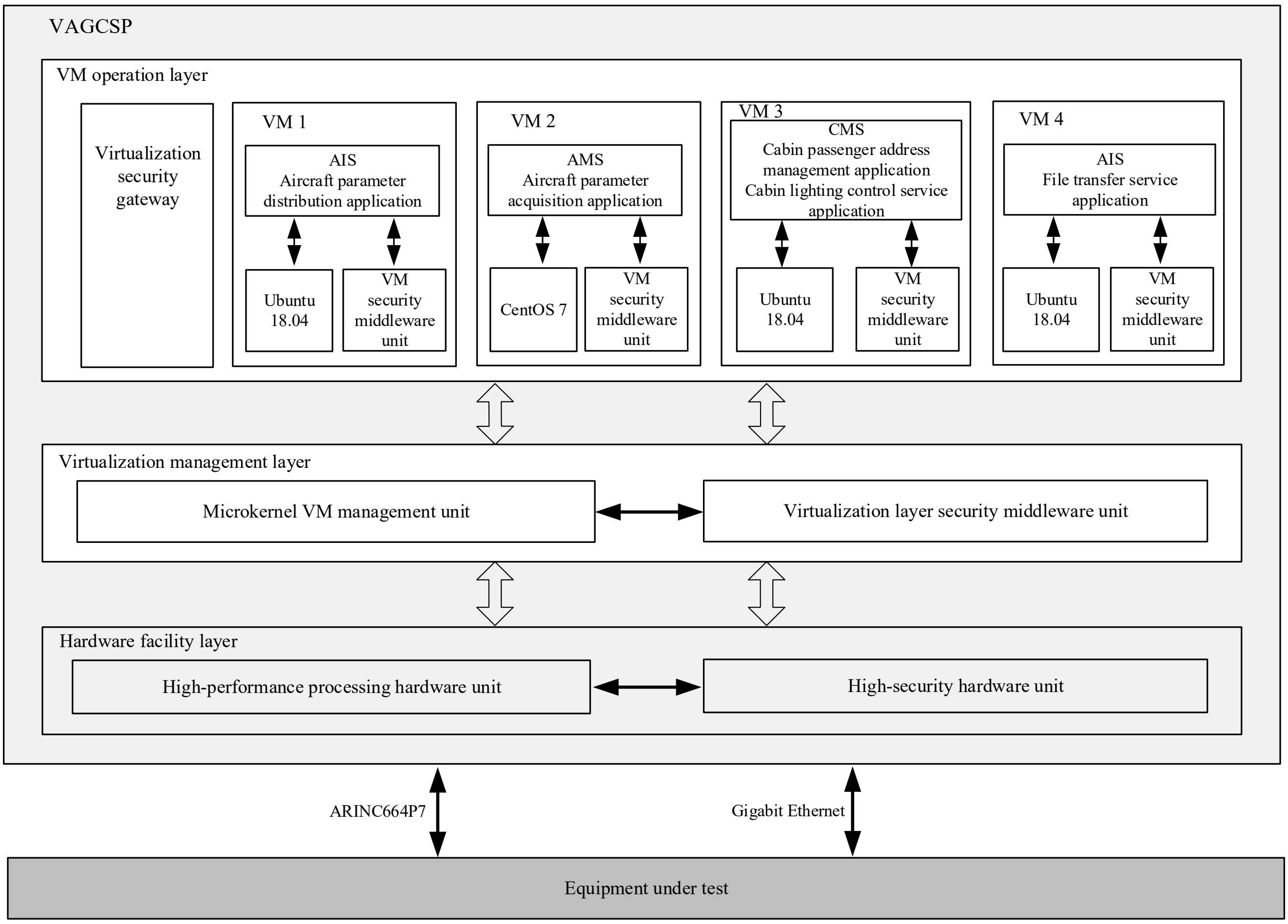
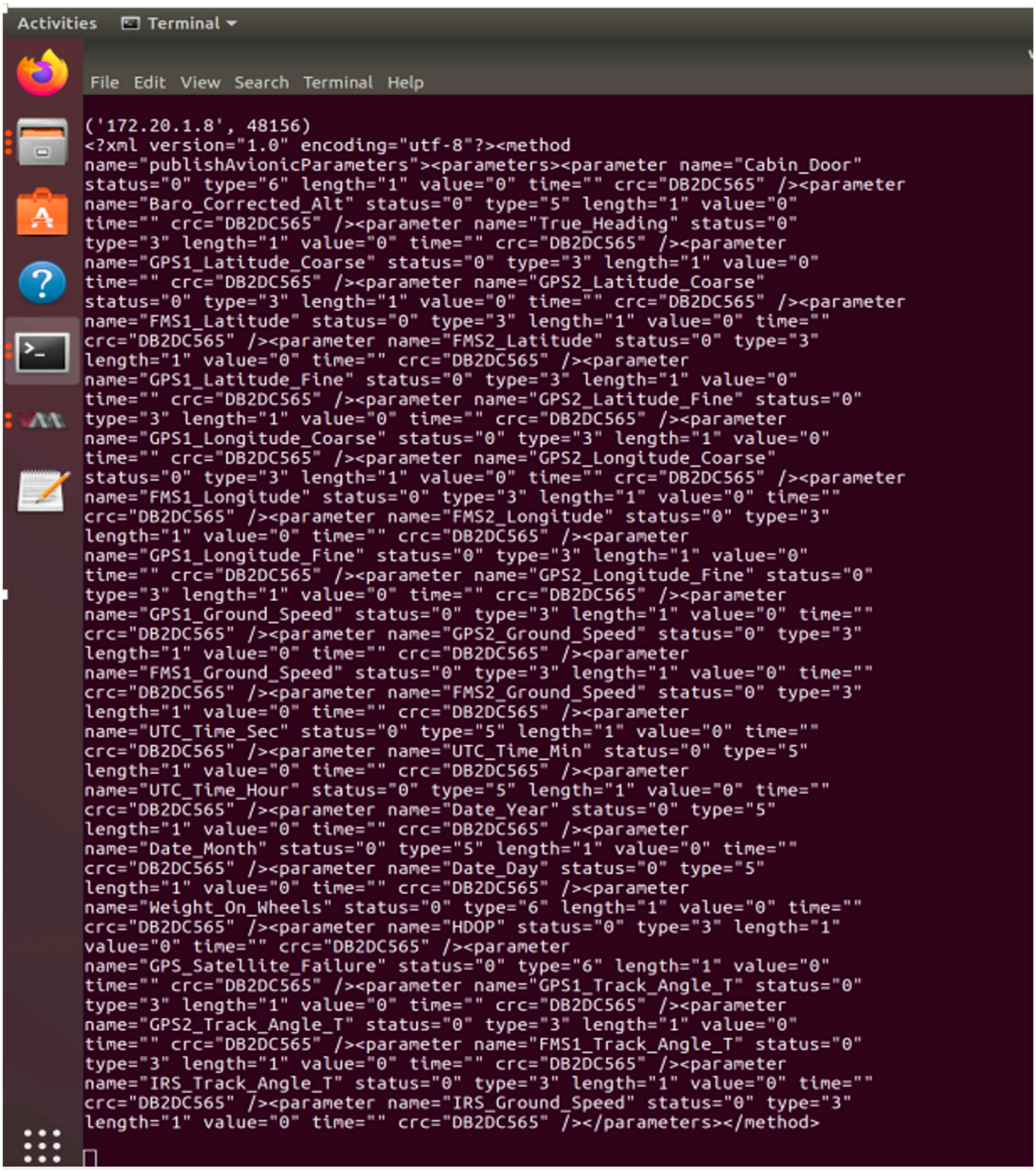

VAGCSP
VM operation layer
Virtualization security gateway
VM 1
AIS
Aircraft parameter distribution application
Ubuntu 18.04
VM security middleware unit
VM 2
AMS
Aircraft parameter acquisition application
CentOS 7
VM security middleware unit
VM 3
CMS
Cabin passenger address management application
Cabin lighting control service application
Ubuntu 18.04
VM security middleware unit
VM 4
AIS
File transfer service application
Ubuntu 18.04
VM security middleware unit
Virtualization management layer
Microkernel VM management unit
Virtualization layer security middleware unit
Hardware facility layer
High-performance processing hardware unit
High-security hardware unit
ARINC664P7
Gigabit Ethernet
Equipment under test

Figure 4. Verification environment of the application hosting function.

Table 1. Test items of the application hosting function.

Systems	Typical Applications	Function Test Items
airborne information system	aircraft parameter distribution application	distributing aircraft parameters
	file transfer service application	downloading files uploading files
airborne maintenance system	aircraft parameter acquisition application	acquiring aircraft parameters and information
cabin management system	cabin passenger address management application	enabling passenger address disabling passenger address
	cabin lighting control service application	turning on the no smoking and fasten seat belt lights at terminal turning off the no smoking and fasten seat belt lights at terminal turning on the reading light turning off the reading light



```
('172.20.1.8', 48156)
<?xml version="1.0" encoding="utf-8"?><method
name="publishAvionicParameters"><parameters><parameter name="Cabin_Door"
status="0" type="6" length="1" value="0" time="" crc="DB2DC565" /><parameter
name="Baro_Corrected_Alt" status="0" type="5" length="1" value="0"
time="" crc="DB2DC565" /><parameter name="True_Heading" status="0"
type="3" length="1" value="0" time="" crc="DB2DC565" /><parameter
name="GPS1_Latitude_Coarse" status="0" type="3" length="1" value="0"
time="" crc="DB2DC565" /><parameter name="GPS2_Latitude_Coarse"
status="0" type="3" length="1" value="0" time="" crc="DB2DC565" /><parameter
name="FMS1_Latitude" status="0" type="3" length="1" value="0" time=""
crc="DB2DC565" /><parameter name="FMS2_Latitude" status="0" type="3"
length="1" value="0" time="" crc="DB2DC565" /><parameter
name="GPS1_Latitude_Fine" status="0" type="3" length="1" value="0"
time="" crc="DB2DC565" /><parameter name="GPS2_Latitude_Fine" status="0"
type="3" length="1" value="0" time="" crc="DB2DC565" /><parameter
name="GPS1_Longitude_Coarse" status="0" type="3" length="1" value="0"
time="" crc="DB2DC565" /><parameter name="GPS2_Longitude_Coarse"
status="0" type="3" length="1" value="0" time="" crc="DB2DC565" /><parameter
name="FMS1_Longitude" status="0" type="3" length="1" value="0" time=""
crc="DB2DC565" /><parameter name="FMS2_Longitude" status="0" type="3"
length="1" value="0" time="" crc="DB2DC565" /><parameter
name="GPS1_Longitude_Fine" status="0" type="3" length="1" value="0"
time="" crc="DB2DC565" /><parameter name="GPS2_Longitude_Fine" status="0"
type="3" length="1" value="0" time="" crc="DB2DC565" /><parameter
name="GPS1_Ground_Speed" status="0" type="3" length="1" value="0" time=""
crc="DB2DC565" /><parameter name="GPS2_Ground_Speed" status="0" type="3"
length="1" value="0" time="" crc="DB2DC565" /><parameter
name="FMS1_Ground_Speed" status="0" type="3" length="1" value="0" time=""
crc="DB2DC565" /><parameter name="FMS2_Ground_Speed" status="0" type="3"
length="1" value="0" time="" crc="DB2DC565" /><parameter
name="UTC_Time_Sec" status="0" type="5" length="1" value="0" time=""
crc="DB2DC565" /><parameter name="UTC_Time_Min" status="0" type="5"
length="1" value="0" time="" crc="DB2DC565" /><parameter
name="UTC_Time_Hour" status="0" type="5" length="1" value="0" time=""
crc="DB2DC565" /><parameter name="Date_Year" status="0" type="5"
length="1" value="0" time="" crc="DB2DC565" /><parameter
name="Date_Month" status="0" type="5" length="1" value="0" time=""
crc="DB2DC565" /><parameter name="Date_Day" status="0" type="5"
length="1" value="0" time="" crc="DB2DC565" /><parameter
name="Weight_On_Wheels" status="0" type="6" length="1" value="0" time=""
crc="DB2DC565" /><parameter name="HDOP" status="0" type="3" length="1"
value="0" time="" crc="DB2DC565" /><parameter
name="GPS_Satellite_Failure" status="0" type="6" length="1" value="0"
time="" crc="DB2DC565" /><parameter name="GPS1_Track_Angle_T" status="0"
type="3" length="1" value="0" time="" crc="DB2DC565" /><parameter
name="GPS2_Track_Angle_T" status="0" type="3" length="1" value="0"
time="" crc="DB2DC565" /><parameter name="FMS1_Track_Angle_T" status="0"
type="3" length="1" value="0" time="" crc="DB2DC565" /><parameter
name="IRS_Track_Angle_T" status="0" type="3" length="1" value="0" time=""
crc="DB2DC565" /><parameter name="IRS_Ground_Speed" status="0" type="3"
length="1" value="0" time="" crc="DB2DC565" /></parameters></method>
```

Figure 5. Test results of aircraft parameter distribution application.

4.2. Platform Performance Test

During the VAGCSP performance verification, the VM performance was mainly tested in terms of the processor, memory, and Ethernet, and the test results were compared with the performance data of an existing AGCSP. The LmBench tool was used to examine the processor performance, with respect to integer calculation, floating-point calculation, file access, and process overheads. Table 2 details the test results. The UDP and TCP communication performance were tested using Iperf3, and Table 3 details the test results.

Table 2. LmBench test results.

Test Categories	Test Items	AGCSP	VAGCSP VM
Integer calculation	intgr bit	0.84 ns	0.25 ns
	intgr div	29.2 ns	9.74 ns
	intgr mod	14.2 ns	10.1 ns
Floating-point calculation	float add	6.83 ns	1.44 ns
	float div	31.9 ns	4.15 ns
	float bogo	45.2 ns	1.08 ns
File access	10k File Create	431.8 us	16.3 us
	10k File Delete	6.0614 us	4.8234 us
	100fd select	5.558 us	1.625 us
Process overhead	sig inst	1.14 us	0.59 us
	fork proc	340 us	137 us
	exec proc	1075 us	472 us
	shproc	3835us	1190 us

Table 3. Iperf3 test results.

Test Categories	Test Items	AGCSP	VAGCSP VM
TCP	Throughput	940.2 Mbits/s	941.8 Mbits/s
	Multithread throughput	961.2 Mbits/s	963.7 Mbits/s
	Uplink/downlink bandwidth	941.2 Mbits/s	944.6 Mbits/s
UDP	Throughput	972.6 Mbits/s	983.9 Mbits/s
	Multithread throughput	985.7 Mbits/s	992.9 Mbits/s
	Uplink/downlink bandwidth	973.7 Mbits/s	984.8 Mbits/s

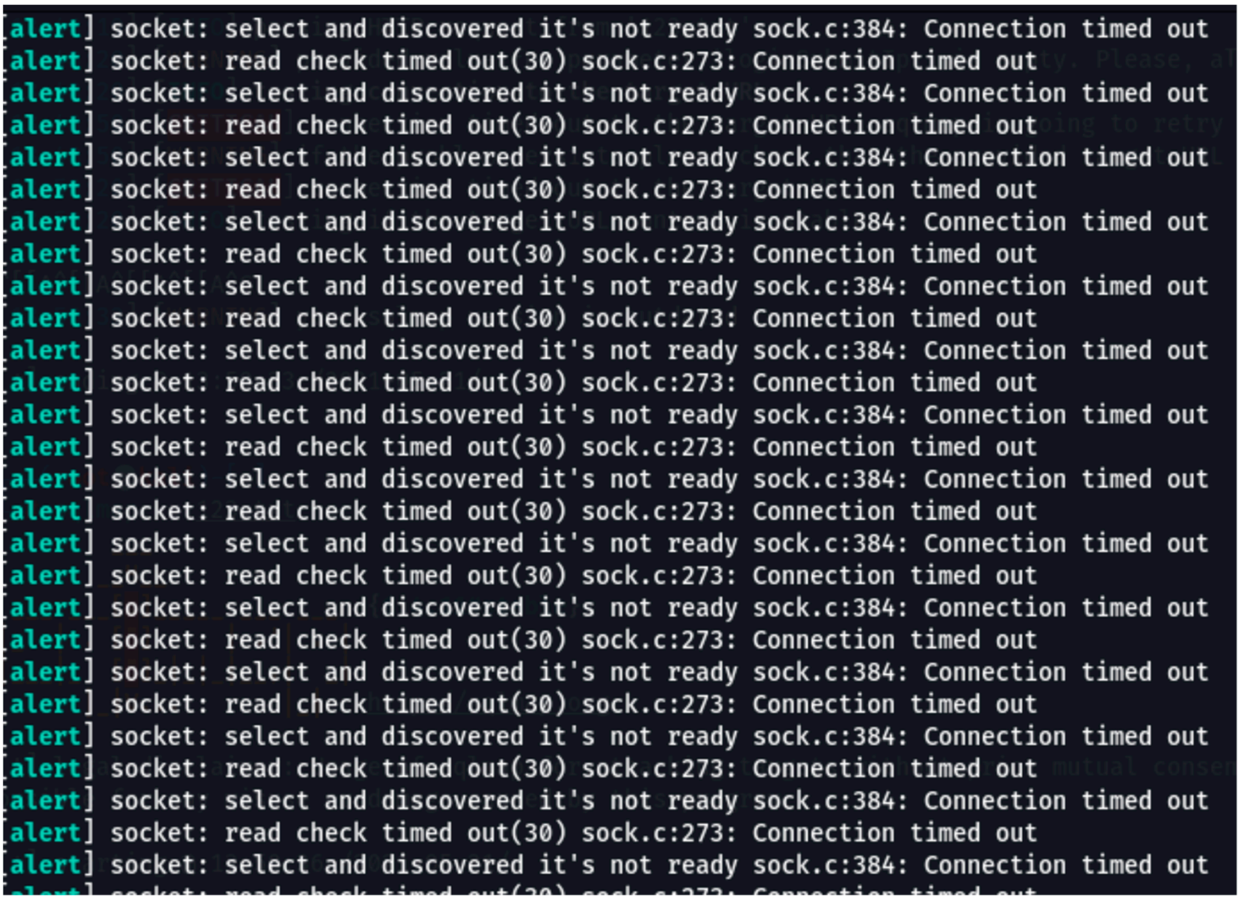
For the four VMs in the verification environment, the encryption algorithm uses AES, and the hash algorithm uses SHA256. Table 4 details the VM startup time test results, which are the average of 20 experiments.

Table 4. VM startup time test results.

	VM1	VM2	VM3	VM4
Size of Image	245.32 Mbytes	202.78 Mbytes	232.66 Mbytes	221.64 Mbytes
Decryption time	1.091 s	0.902 s	1.035 s	0.986 s
Trusted measurement	0.772 s	0.638 s	0.732 s	0.698 s
OS startup	6.213 s	5.568 s	6.021 s	5.735 s
Total time	8.076 s	7.108 s	7.788 s	7.418 s

4.3. Verification of Network Security Protection Capabilities

In the verification environment for the network security protection capabilities, airborne network security attack testing equipment was added on the basis of the application hosting function verification environment in Figure 4. The equipment was connected to the VAGCSP via a one-channel aviation Ethernet interface, which executed network security tests such as a DDoS attack, worm attack, backdoor attack, and vulnerability scanning and access control penetration testing by sending attack packets to VAGCSP, thereby verifying VAGCSP's network security protection capabilities. The experimental steps were as follows: (1) Trusted loading function of VMs was tested. (2) Application function test was implemented on VAGCSP while sending network attack packets to it. (3) CVE vulnerability scanning test was executed on VAGSCP. (4) Unauthorized network access attack was executed against VAGSCP. The experimental results were examined by checking the output of the test case results running on the testing equipment, as well as the log records of VAGCSP. The experimental results showed that (1) VAGSCP correctly implemented the measurement and loading of VMs. (2) After the injecting network attack, the application functions of VAGCSP were executed correctly. (3) VAGCSP had no vulnerabilities above the low-risk level. (4) VAGSCP could effectively block the unauthorized network access requests, and Figure 6 displays the test results.



```
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
[alert] socket: select and discovered it's not ready sock.c:384: Connection timed out
[alert] socket: read check timed out(30) sock.c:273: Connection timed out
```

Figure 6. Results of the unauthorized network access test.

4.4. Analysis of the Experimental Results

According to the results of application hosting function verification, the VAGCSP provides high-performance processing capacity with its four-core eight-thread processor and mass storage capacity with 4 TB SSD, integrated one 100 Mbps ARINC664P7 interface, and six gigabit Ethernet interfaces. VAGCSP supports simultaneous residence of avionics system applications with low safety levels, such as airborne information systems, airborne maintenance systems, and cabin management systems, which can provide a secure and isolated operating environment for applications in each VM.

The results of the platform performance verification show that compared to the existing platform, the VAGCSP VMs have greatly improved performance in terms of integer calculation, floating-point calculation, and process overheads. Regarding the file access performance, the VAGCSP performed well in file creation and file handle serial processing; meanwhile, regarding the file deletion performance, the two platforms exhibited similar performance parameters, since both of them adopt SSDs. Moreover, little differences were noted in the network transfer performance data between the two platforms, since both of them use gigabit Ethernet interfaces, with the VAGCSP VMs slightly outperforming the other platform. While the weight and power consumption of the VAGCSP are slightly less than those of AGCSP devices, the computing performance of the VAGCSP is improved by over one-fold.

As revealed by the verification of its network security protection capabilities, the VAGCSP can offer trusted loading of VMs and provide them with security protection covering east–west traffic access control, intrusion detection and defense, malicious code detection, etc. With a high-security hardware unit, VM security middleware unit, and virtualization security gateway, the VAGCSP provides a secure and trusted computing environment.

Based on these experiment results, VAGCSP completely achieved the design objectives.

5. Conclusions

Responding to the multifarious requirements of new-generation civil aircraft, such as energy conservation, emission reduction, high-performance computing, and information security protection, a virtualization airborne trusted general computing service architecture was proposed, and key technologies, including high-performance processing and high-security hardware units, a virtualization management software unit, and virtualization security protection framework, were designed. Based on the physical prototype of the VATGCSM, application hosting function, platform performance, and network security protection experiments were carried out. The experimental verification revealed that the proposed method could meet the technical requirements and is applicable, not only to new-generation civil aircraft, but also to UAVs and emergency rescue aircraft with high-performance safety-critical computing needs.

Author Contributions: Methodology, S.Z.; Hardware and software, S.Z., Y.W. and X.W.; Validation, S.Z., Y.W. and Z.L.; Writing—original draft preparation, S.Z.; Writing—review and editing, S.Z.; Supervision, Y.G.; Project administration, Y.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by National Key Scientific Research Project (No. MJ-2018-S-34).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the reviewers and the editor for their comments and suggestions, which helped to improve the paper significantly.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. *Arinc 763A*; Mark 2 Networks Server System (NSS) form and Fit Definition. Aeronautical Radio, Inc.: Annapolis, MD, USA, 2008; pp. 3–8.
2. *Arinc 821*; Aircraft Network Server System (NSS) Functional Definition. Aeronautical Radio, Inc.: Annapolis, MD, USA, 2008; pp. 4–9.
3. Liu, X.; Li, L.; Zhang, S.; Zhang, J.; Zhang, T. Design and Implementation of an Avionics Interface Application Software for Onboard Network Service System. *Electron. Opt. Control* **2015**, *2*, 70–74.
4. *Do-326A*; Airworthiness Security Process Specification. RTCA, Inc.: Washington, DC, USA, 2014; pp. 7–15.
5. *Do-356A*; Airworthiness Security Methods and Considerations. RTCA, Inc.: Washington, DC, USA, 2018; pp. 23–28.
6. Zhang, S.; Kong, D.; Wang, Y.; Wan, X.; Yao, H.; Guo, Y. Secure communication technology between network domains based on virtualization avionics platform. *J. Northwest. Polytech. Univ.* **2022**, *40*, 530–537. [CrossRef]
7. Chen, G.; Guan, N.; Lu, M.; Wang, Y. State-of-the-Art Survey of Real-Time Multicore System. *J. Softw.* **2018**, *29*, 2152–2176.
8. Zhang, S.; Wan, X.; Kong, D.; Guo, Y. Embedded Virtualization Computing Platform Security Architecture Based on Trusted Computing. In Proceedings of the 2020 7th International Conference on Dependable Systems and Their Applications (DSA), Xi'an, China, 28–29 November 2020; pp. 231–235.
9. Zhang, X.; Wang, Y. DeepMECagent: Multi-agent computing resource allocation for UAV-assisted mobile edge computing in distributed IoT system. *Appl. Intell.* **2022**, *53*, 1180–1191. [CrossRef]
10. Zhang, J.; Letaief, K.B. Mobile edge intelligence and computing for the internet of vehicles. *Proc. IEEE* **2020**, *108*, 246–261. [CrossRef]
11. Mallasch, P.G.; Miller, B.; Schramm, J. Platform as a Service (PaaS) as an Alternative for Commercial Aviation Applications. In Proceedings of the 2013 Aviation Technology, Integration, and Operations Conference, Los Angeles, CA, USA, 12–14 August 2013.

12. Wang, B.; Xie, J.; Li, S.; Wan, Y.; Fu, S.; Lu, K. Enabling high-Performance Onboard Computing with Virtualization for Unmanned Aerial Systems. In Proceedings of the 2018 International Conference on Unmanned Aircraft Systems (ICUAS), Dallas, TX, USA, 12–15 June 2018; pp. 202–211.
13. Jiang, Z.; Parimi, A. A Real-Time Computing Platform for UAS System Dynamics and Control Simulation. In Proceedings of the AIAA Propulsion and Energy 2021 Forum, Virtual, 9–11 August 2021.
14. Smagin, D.I.; Savelev, R.S.; Satin, A.A. Methods for the Design of Modern On-Board Systems of Advanced Aircraft. In Proceedings of the 2019 IEEE 10th International Conference on Mechanical and Aerospace Engineering (ICMAE), Brussels, Belgium, 22–25 July 2019; pp. 97–101.
15. Douklias, A.; Karagiannidis, L.; Misichroni, F.; Amditis, A. Design and implementation of a UAV-based airborne computing platform for computer vision and machine learning applications. *Sensors* **2022**, *22*, 2049. [CrossRef] [PubMed]
16. Jayaram Masti, R.; Marforio, C.; Capkun, S. An Architecture for Concurrent Execution of Secure Environments in Clouds. In Proceedings of the 2013 ACM Workshop on Cloud Computing Security Workshop ACM, Berlin, Germany, 8 November 2013; pp. 11–22.
17. Sun, Z.; Feng, B.; Lu, L.; Jha, S. OAT: Attesting Operation Integrity of Embedded Devices. In Proceedings of the 2020 IEEE Symposium on Security & Privacy, San Francisco, CA, USA, 18–21 May 2020; pp. 1–17.
18. Elmiligi, H.; Gebali, F.; El-Kharashi, M.W. Multi-dimensional analysis of embedded systems security. *Microprocess. Microsyst.* **2016**, *41*, 29–36. [CrossRef]
19. Mansouri, Y.; Babar, M.A. A review of edge computing: Features and resource virtualization. *J. Parallel Distrib. Comput.* **2021**, *150*, 155–183. [CrossRef]
20. Cinque, M.; Cotroneo, D.; De Simone, L.; Rosiello, S. Virtualizing mixed-criticality systems: A survey on industrial trends and issues. *Future Gener. Comput. Syst.* **2022**, *129*, 315–330. [CrossRef]
21. Lata, S.; Singh, D. Intrusion detection system in cloud environment: Literature survey & future research directions. *Int. J. Inf. Manag. Data Insights* **2022**, *2*, 100134.
22. Asvija, B.; Eswari, R.; Bijoy, M.B. Security in hardware assisted virtualization for cloud computing—State of the art issues and challenges. *Comput. Netw.* **2019**, *151*, 68–92. [CrossRef]
23. Compastié, M.; Badonnel, R.; Festor, O.; He, R. From virtualization security issues to cloud protection opportunities: An in-depth analysis of system virtualization models. *Comput. Secur.* **2020**, *97*, 101905. [CrossRef]
24. Cotroneo, D.; De Simone, L.; Natella, R. Timing covert channel analysis of the VxWorks MILS embedded hypervisor under the common criteria security certification. *Comput. Secur.* **2021**, *106*, 102307. [CrossRef]
25. Yang, Y.; Jiang, H.; Zhang, G.; Wang, X.; Lv, Y.; Li, X.; Serge, F.; Xie, G. S2H: Hypervisor as a setter within Virtualized Network I/O for VM isolation on cloud platform. *Comput. Netw.* **2021**, *201*, 108577. [CrossRef]
26. Pakmehr, M.; Khamvilai, T.; Behbahani, A.R.; Costello, J.; Skertic, R.; Ademola, A.P. Applying Zero Trust Principles to Distributed Embedded Engine Control Systems. In Proceedings of the AIAA Aviation 2022 Forum, Chicago, IL, USA, 27 June–1 July 2022.
27. Yarza, I.; Agirre, I.; Mugarza, I.; Cerrolaza, J.P. Safety and security collaborative analysis framework for high-performance embedded computing devices. *Microprocess. Microsyst.* **2022**, *93*, 104572. [CrossRef]
28. Bottaro, M.; Vardanega, T. Evaluating a multicore Mixed-Criticality System implementation against a temporal isolation kernel. *J. Syst. Archit.* **2022**, *130*, 102688. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.