Article

Will Zero Vulnerability Computing (ZVC) Ever Be Possible? Testing the Hypothesis

Fazal Raheman [1,*], Tejas Bhagat [1], Brecht Vermeulen [2] and Peter Van Daele [2]

[1] Blockchain 5.0 Ltd., Kesklinna Linnaosa, Ahtri tn 12, 10151 Tallinn, Estonia; tejas@bc5.eu
[2] IDLab iGent Tower—Department of Information Technology, IMEC—Ghent University, Technologiepark-Zwijnaarde 126, B-9052 Ghent, Belgium; brecht.vermeulen@ugent.be (B.V.); pet.vandaele@ugent.be (P.V.D.)
* Correspondence: drfazal@bc5.eu

Abstract: Life without computers is unimaginable. However, computers remain vulnerable to cybercrimes, a USD 6 trillion industry that the world has come to accept as a "necessary evil". Third-party permissions resulting in an attack surface (AS) and in-computer storage that computers mandate are key design elements that hackers exploit, formerly by remote malware installation and later by stealing personal data using authentication faking techniques. In legacy computers, the AS cannot be completely eliminated, nor can a connected device retain data offline, rendering fool-proof cybersecurity impossible. Although the architects of legacy computers made perfectly reasonable engineering trade-offs for their world, our world is very different. Zero vulnerability computing (ZVC) challenges the impossible with in-computer offline storage (ICOS) and Supra OS (SOS), to deliver comprehensive protection against vulnerabilities. The feasibility of ZVC is demonstrated in a tiny permanently computer-mounted hardware wallet, providing the first evidence of the complete obliteration of the AS. Malware cannot infect the ZVC device on account of lacking an AS, nor can personal data be hacked as they mostly remain offline, except for sporadic processing. Further research should explore whether ZVC can fully secure computers in more complex real-world scenarios and open a new epoch in the evolution of computers and the Internet.

Keywords: cybercrimes; cyber security; computer vulnerabilities; attack surface; IoT; computer architecture; malware; hacking

check for updates

Citation: Raheman, F.; Bhagat, T.; Vermeulen, B.; Van Daele, P. Will Zero Vulnerability Computing (ZVC) Ever Be Possible? Testing the Hypothesis. *Future Internet* **2022**, *14*, 238. https://doi.org/10.3390/fi14080238

Academic Editor: Paolo Bellavista

Received: 28 June 2022
Accepted: 28 July 2022
Published: 30 July 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Cybercrime inflicts damages totaling USD 6 trillion annually and is predicted to grow to over USD 10.5 trillion industry by 2025 [1]. A hack attack occurs **every 39 s,** and approximately **300,000** new malware programs are created daily [2]. Today, over 4.5 billion connected devices remain at risk of cyber-attack [3]. With the exponential proliferation of IoT devices [4] and the ever-growing vulnerabilities of connected devices, the cybercrime industry is poised for unstoppable growth [5].

Advances in computer science and IoT computing have been running at ultrahigh speeds. We are looking at the future in which everything will become a connected computing device [6]. Toys, clothes, cars, door locks, contact lenses, clothes, toasters, refrigerators, washing machines, light bulbs, toothbrushes, glasses, helmets, etc., are getting "smart". Everything has become a branch of computer science.

"Everything is now a computer. The digital nightmare is about to get much worse", claims Bruce Schneier in his latest book ***Click Here to Kill Everybody*** [7].

A Brief History of Computers & the Origin of Their Vulnerabilities

Back in the 1940s, Thomas Watson, boss of the giant IBM Corporation, made probably the worst prediction in history, "the world would need no more than about five computers [8]. Eight decades later, experts are predicting 50 billion Internet-connected devices by 2030 [9].

The first computers were gigantic calculating machines, and all they ever really did was "crunch numbers": solve lengthy, difficult, or tedious mathematical problems. There was no software industry; no operating systems and programs were custom written for each computer individually until IBM launched its first personal computer in 1981 with an operating system that granted third-party permissions to developers of software applications. Since then, it became standard practice to produce computers with third-party permissions. In the 1990s, with the advent of the Internet, bad actors started exploiting these permissions to write malicious programs and install them on victim computers remotely. Such attack vectors created an attack surface that introduced vulnerabilities, feeding a thriving multitrillion cybercrime industry. In the state-of-the-art, it is impossible to build a computer without third-party permissions and consequently free of vulnerabilities. Essentially, legacy computers cannot prevent malware from exploiting the inherent permissions that are available for all third-party programs. However, the network efficiencies have reached a level that even the most resource-intensive applications can now run remotely without the need for local installation [10]. This introduces the possibility of doing away with third-party permissions entirely and, consequently, eliminating the attack surface to achieve zero vulnerability. Our research on zero vulnerability computing exploits these possibilities for rendering computing devices immune to cyber-attacks.

2. Problem Statement

Experts unanimously agree that fool-proof cybersecurity is "practically impossible". This is essentially because of the following two mandatory design attributes that legacy computers must comply with to render them usable:

- **Permissions** that third-party applications need to run, but bad actors often exploit to introduce vulnerabilities, resulting in attack surfaces that can be exploited by malware [11].
- **In-computer data storage,** in absence of which, using, processing, or saving data is impossible. However, because most computers remain connected, stored data are continuously exposed to hackers. Hence, any data stored in a networked computing device are considered a priori at risk.

Both **necessary evils** are the core enablers of cybersecurity breaches. State-of-the-art cybersecurity techniques are predominantly focused on strategies that reduce the attack surface and encrypt the data stored in online devices to counter these evils. However, these approaches have known limitations and are often complex, making cybersecurity experts conclude that fool-proof cybersecurity is impossible. Gene Spafford, a National Cybersecurity Hall of Famer asserts:

> *"The only truly secure system is one that is powered off, cast in a lock of concrete and sealed in a lead-lined room with armed guards—and even then, I have my doubts."* [12]

Cybersecurity experts unanimously agree (*"this is a world in which the promise of secure digital technology turns out to be in many respects a poisoned chalice'."*) [13] that data within a connected device can never be entirely secure, because network exposure can never be risk-free [14].

Since the birth of the Internet and the cybercrime industry, the attack surface has been growing. The severity of vulnerability or common vulnerabilities and exposures (CVEs) is also increasing. The National Vulnerability Database (NVD) [15] provides qualitative severity rankings in the scale [*Low-Medium-High*] depending on the Common Vulnerabilities Scoring System (CVSS) as defined in the CVSS specification [16], which scores from 0–10 in an increasing order of severity. Of the top 50 products [17] reporting the total number of distinct vulnerabilities in 2020, OSs were directly or indirectly responsible for almost all reported vulnerabilities.

2.1. Perpetually Increasing Attack Surface

Recent reports forecast a surge of 12.6% in CAGR by 2027. With the proliferation of IoT devices, predicted to reach 75 billion by 2025 [18] (Figure 1), the attack surface will grow exponentially, increasing security vulnerabilities across the board. This extraordinary growth of a serious problems warrants extraordinary measures. Is our current cybersecurity strategy sufficient? Let us review them.

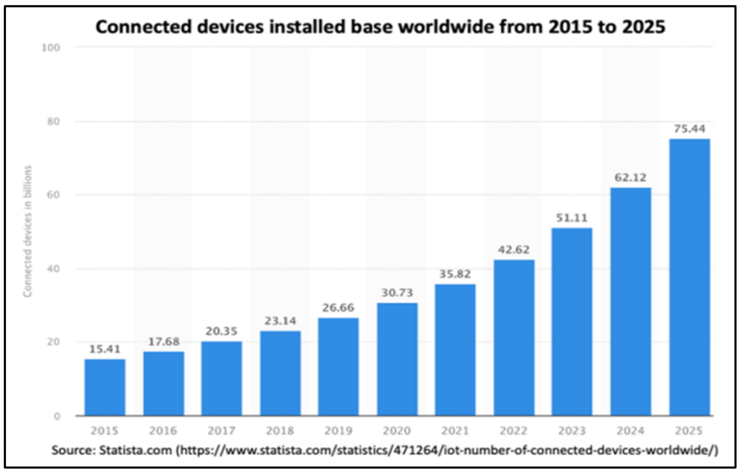


Figure 1. The exponential growth of connected devices (Statista.com).

"Complexity is the worst enemy of security, and this is especially true for computers & the Internet." [19]. Complexity opens the door to vulnerability. Unfortunately, advancements in computer technology have always been associated with increasing complexity. The more complex a system is, the more vulnerable it is to security threats. It is time we reversed this trend and eliminated all complexities ingrained in legacy systems. With the advent of the IoT and the proliferation of connected devices, attack surfaces and, consequently, vulnerabilities have grown exponentially. Every day, the AV-TEST Institute registers over 450,000 new malicious programs (malware) and potentially unwanted applications (PUA). Over the past decade, the number of malware cases has grown from approximately 100 million in 2012 to a whooping 1.33 billion in 2021 [20]. See Figure 2.

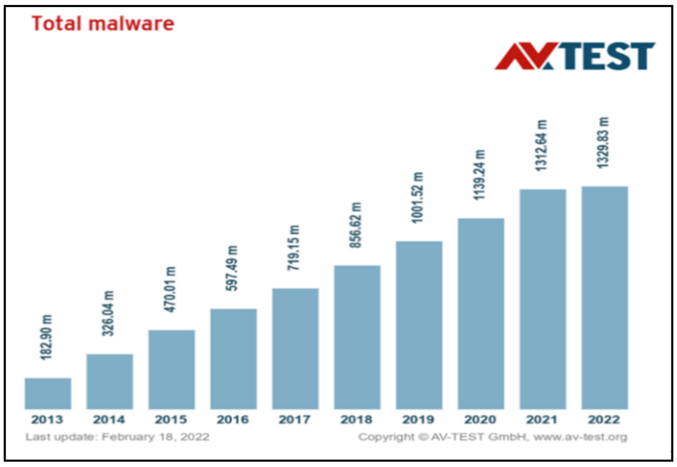


Figure 2. Every day, the AV-TEST Institute registers over 450,000 new malicious programs (malware) and potentially unwanted applications (PUA). Of total threats examined, 95.30% were malware and 4.70% were PUAs (source: https://www.av-test.org, accessed on 27 June 2022).

In the current state-of-the-art, the approach to improving the security of a computer system is to measure the attack surface [21] and minimize it using the following basic strategies:

- Reducing the amount of code running;
- Reducing entry points available to untrusted users;
- Eliminating services requested by relatively few users [22].

The zero-trust architecture by NIST also suggests a similar strategy [23]. Although attack surface reduction helps prevent many security failures, it does not mitigate the damage an attacker can inflict once a software vulnerability is found [24].

2.2. Increasing Zero-Day Vulnerabilities

Code reuse is a common practice in software development owing to its various benefits [25]. The use of both commercial and open-source components in development increases the vulnerability. However, such a practice causes large-scale security issues because one vulnerability may appear in different software because of cloned code fragments [26]. This rising trend has become a major reason for the constantly expanding attack surface in the past decade. Zhang et al. (2019) [27] recently demonstrated that many seemingly unrelated software apps share a common attack surface. In addition to the software development trends that increase the attack surface, the timeframe of vulnerability exploitation is compressed by 93%. Currently, it is only three days before a vulnerability is exploited, compared to 45 days in 2006 [28]. Cybersecurity ventures [29] predicted zero-day cyber-attacks to rise from one per week to one per day.

2.3. The Q-Day Threat

Last but not the least, two recent articles in the *Nature* journals emphasize the seriousness of the impending threats from quantum computers to the Internet [30] and an actual quantum attack on several cryptocurrencies that led to the latest crypto crash of 2022 [31]. Already overwhelmed with the ever-increasing scourge of computer vulnerabilities and cybercrimes, the cybersecurity of computers appears to be moving closer to the apocalyptic threats from quantum computers [32].

3. State-of-the-Art

Our problem statement identifies two "necessary evils" as the core enablers of cybersecurity breaches in legacy computing systems. We designed a zero vulnerability computing (ZVC) solution that tackles each of those design mandates that make computers usable, but paradoxically render them vulnerable. We define ZVC as a new cybersecurity paradigm that proposes a new computer architecture.

A review of the state-of-the-art therefore warrants a two-fold inquiry.

3.1. Offline Storage Technologies

In legacy systems, offline storage refers to any storage medium that must be physically inserted into a system every time a user wants to access or edit data. Offline storage can be any type of storage that can be easily removed from the computer and is not currently online, live, or connected to the computer. The data stored in offline storage remain permanently in the storage device even if it is disconnected or unplugged from the computer after the data have been stored. Offline storage is generally portable in nature and can be used on different computer systems. Known also as removable storage, examples of offline storage include floppy disks, compact disks, and USB sticks.

In computers of the prior state-of-the-art, data storage constitutes one of the key components of the computer architecture. However, there is no provision or concept of offline storage within a connected computer. In other words, when a computer connects to the Internet, the stored data, by default, also go online, exposing it to network risks. Hence, the state-of-the-art is completely blank in terms of the in-computer offline storage (ICOS) module that constitutes one of the two modules of the ZVC architecture.

3.2. Technologies for Secure Execution Environments

To cope up with the ever-increasing demand in hardware and provide solutions to the exploitation of vulnerabilities in program execution, libraries, and the operating system APIs, different kinds of architectures such as sandboxes, virtual machines, and containerized application systems have emerged (Figure 3).

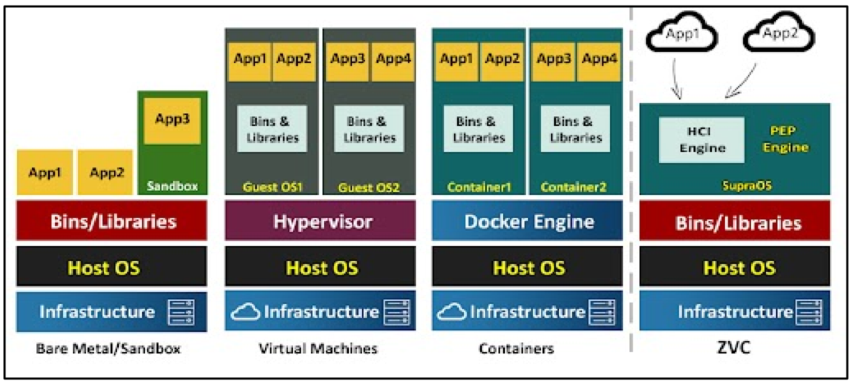


Figure 3. State-of-the-art cybersecurity approaches compared with ZVC's zero attack surface approach.

Sandboxing was the first practice to establish the concept of running code in a safe, isolated environment that mimics end-user OS APIs. Sandboxes run as software in the application layer of existing OSs and execute potentially dangerous system calls inside a controlled environment. Sandboxes are inefficient in deploying complicated security policies, do not scale, and cannot protect the OS from various existing monitor-bypassing attacks.

Virtual machines (VMs) are the typical building blocks for cloud infrastructures to gain higher hardware resource utilization while preserving execution isolation among different computing instances [33]. A host OS is isolated through a hypervisor engine, and guest OSs are run in a protected environment. Execution within a guest OS only affects the guest OS API and its underlying software without ever reaching the host OS and its infrastructure. Still, from an execution perspective, having a dedicated OS for each VM and the full set of underlying software suggests that the attack surface and relevant security aspects for the VMs themselves do not differ from that of physical machines [34]. On top of that, various attacks exist that can escape the isolated environment and attack the host OS by leveraging the abundant resources found within the guest OS of a VM [35].

Containerized applications, such as Docker and Linux Containers (LXC), emerged as an alternative to VMs by providing a convenient way to deploy and isolate applications without installing a guest OS and relevant auxiliary software necessary in a VM. Since there is no guest OS virtualization in a container, the attack surface bound to each guest OS and its software services cannot be exploited. However, as containers share the same underlying host kernel, access to containers is not easily secured, and even though Docker isolates parts of the underlying host OS API from containerized software, this can be an issue for multi-tenant security [36].

Air gapping: Although sandboxing, VMs, and containerization provide a high level of protection, if the local area network is connected to the Internet, a motivated adversary will find a way to breach the network, evade security mechanisms, access sensitive data, and transfer them outside to the attacker [37]. In order to provide protection from such breaches, organizations often store their sensitive data on air-gapped networks. In this approach, any type of physical or logical connection between the local network and the Internet is strictly banned [38]. Air-gapped networks are widely used in military and defense systems [39], critical infrastructure, the finance sector, and other industries [38]. Two famous examples of air-gapped networks are the NSANet and the Joint Worldwide Intelligence Communications System (JWICS), classified (top secret) networks belonging to the United States' Defense Intelligence Agency. Even air-gapped computers caged within an all-metal Faraday cage can be breached as demonstrated by Guri et al. deploying ODINI malware that uses magnetic waves for data transfer [40]. Besides these major cybersecurity approaches, there are several authentication-focused piecemeal protocols such as secure elements, secure tokens, hardware security, trusted platform modules, etc., but none of them has the potential to achieve zero vulnerability whether by completely obliterating the attack surface or creating in-computer offline storage.

3.3. Conclusions

"Complexity is the worst enemy of security, and this is especially true for computers & the Internet." [19]. Complexity opens the door to vulnerabilities. Unfortunately, advancement in computers has always been associated with an increase in complexity. The more complex a system, the more vulnerable it becomes to security threats. It is time we reversed the trend and got rid of all the complexities ingrained in the legacy systems.

4. Research Purpose and Related Works

The principal objective of this research is to explore the feasibility of a conceptual computer architecture framework that revisits the conventional computer design (i.e., the Harvard-like design of microcontrollers and embedded systems or the von Neumann architecture of mainstream computers) that introduces vulnerabilities at all levels of any computing environment. Our primordial efforts to achieve zero vulnerability computing date back two decades, when we first attempted to circumvent operating system vulnerabilities by running a fully quarantined transaction using a CD-ROM device as a real online card key (ROCK) [41]. Although it circumvented the host OS vulnerabilities (spy programs), ROCK was not scalable, limited to specific transactions, and lacked acceptable user experience. This was essentially because web technologies to minimize reliance on native applications were not adequately developed at the time. However, ROCK disclosure provided foundational data to evolve and relate the zero vulnerability computing (ZVC) hypotheses to a computer's attack surface and subsequently to in-computer offline storage (ICOS). Therefore, we formulated two ZVC hypotheses and tested them in experiments that this paper discloses in the following sections: Section 5 provides the rationale of the hypotheses and their verbiage and explanation (A) and defines the innovation (B). Section 6 presents the objectives and details of the experimental setup for supporting and testing the hypotheses. Section 7 discusses the results and conclusions;. Section 8 discusses the work limitations, and Section 9 discusses the future of ZVC.

5. ZVC Hypotheses: New World, New Rules

The architects of legacy computer systems have made perfectly reasonable engineering trade-offs for their world, but our world is very different. The rules governing the computers of their world may no longer be defensible in the hyperconnected world that we are currently living in, at least not considering the pace at which cybersecurity continues to worsen. Therefore, we decided to take a fresh look at the architecture of computers, as it has evolved over time, and start from ground zero. Foundationally, computers were designed oblivious of the serious consequences of the permissions and privileges they afford third-parties, which can potentially make computers vulnerable to threats from bad actors. Such incognizance introduced two mandatory design attributes into the computer architecture:

- Third-party permissions;
- In-computer data storage.

These are foundational elements of the computer architecture, without which prior state-of-the-art computers could not be built. These designs not only introduce complexities, but bad actors also often exploit privileges to create vulnerability. As exploitation techniques improve, the complexities of defense also increase, resulting in a cybersecurity race that has become unwinnable. To explore the possibilities of achieving zero vulnerability computing (ZVC) and winning the race, we conceptualized two radical approaches in redesigning the computer architecture, and to explore the possibilities, we formulated the following hypotheses:

5.1. The ZVC Hypotheses

Hypothesis 1: *Can* ***Supra OS (SOS)*** *completely obliterate the primary attack surface by denying third-party permissions to all non-native applications?*

Hypothesis 2: *Can a hardware module isolate critical data that require sporadic access to offer* ***in-computer offline storage (ICOS)*** *within the connected device itself without compromising its functionality?*

As stated earlier, it is impossible to build a computer without third-party permissions. Permissions create vulnerabilities, allowing attack surfaces that attack vectors use to breach computers. An "attack surface" is essentially the entire externally facing area of a computing environment. It contains all the vulnerabilities or attack vectors that a hacker can use to gain access to a computing system. The attack surface is simply the total digital resource exposed to threats across the enterprise. There is no legacy hardware or software that does not come without third-party permissions by default. As illustrated in Figure 4a–d, third-party permissions may be exploited at the hardware, firmware, or OS level (primary attack surface) or even via the application layer (secondary attack surface). The magnitude of the attack surface is directly proportional to the density of vulnerabilities [42]. The attack surface (represented in Figure 4 by dents or pits and permissions by represented by bumps) is mandatorily created in legacy computers because of the assumption that third-party applications must run locally; therefore, permissions are mandatory. As such, banning all permissions and completely obliterating the attack surface of a computing device are de facto impossibilities in cybersecurity research. However, in the era of 5G communication and ultrafast data transmission speeds, the assumption that the application must be executed locally is outdated and misplaced. Currently, there is no conceivable applications that cannot run on a remote server.

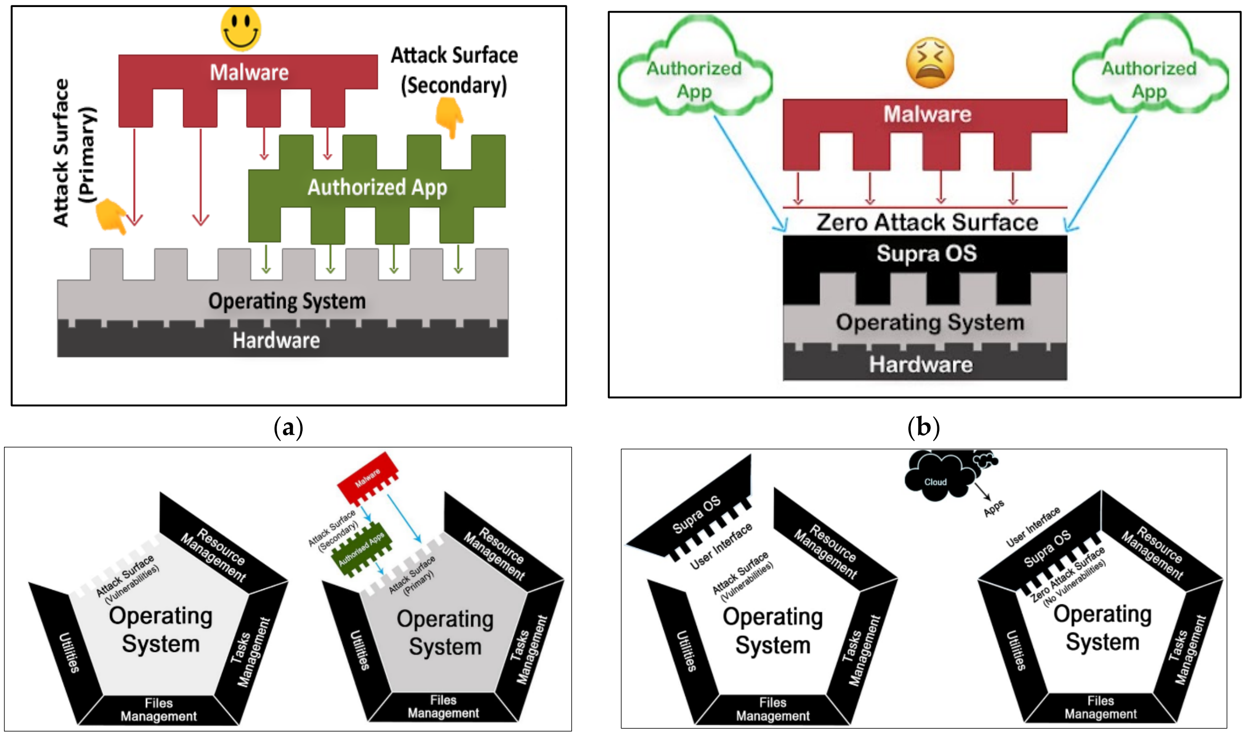
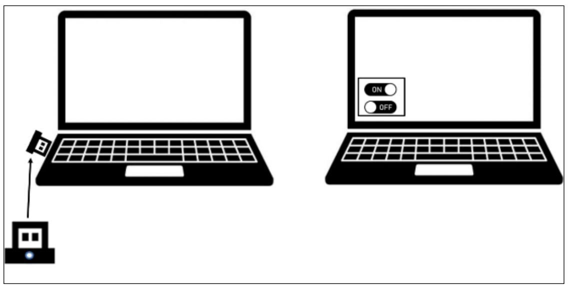
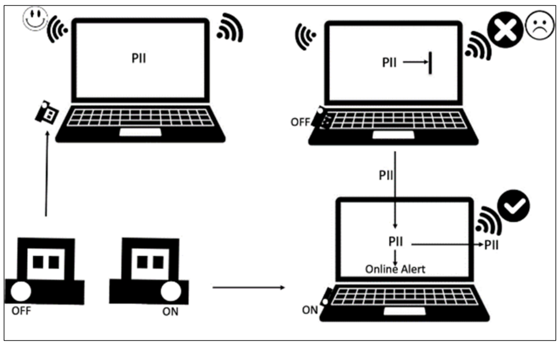


Figure 4. *Cont.*



(**e**)



(**f**)

Figure 4. (**a**) In the traditional computing system, the third-party permissions create primary and secondary attack surfaces that bad actors exploit to breach the computer. (**b**) Zero vulnerability computing (ZVC) completely obliterates the attack surface to achieve a zero attack surface. (**c**) Conceptual diagrammatic representation of primary and secondary attack surfaces. (**d**) Diagrammatic representation of SOS obliterating the attack surface and providing a web interface. (**e**) ICOS with a soft switch and LED indicator. (**f**) ICOS operation with a hard toggle switch.

While banning all third-party permissions to reduce the attack surface to zero may be quite a stretch to conceive of in the pre-5G era, it is quite surprising to note that the state-of-the-art at no point in the evolution of computers made it impossible to integrate a switchable in-computer offline storage (ICOS) memory to complement the computer's standard hard drive. Yet, our legacy computers lack such a mechanism to keep sensitive data offline at users' behest, while the users' computers remain online.

5.2. The Innovation

Considering these realities of the prior state-of-the-art, our SOS hypothesis challenges the very foundation of legacy computer design by banning all permissions for native installation/execution of third-party applications, while allowing only remotely served applications, as illustrated in Figures 4b,d and 5, via a built-in web interface that does not in any way depend on nor communicate with the host computer's operating system. In other words, the ZVC ecosystem runs on its own: it is self-sustaining, running on its own minimalist operating system and has no dependencies on the host operating system.

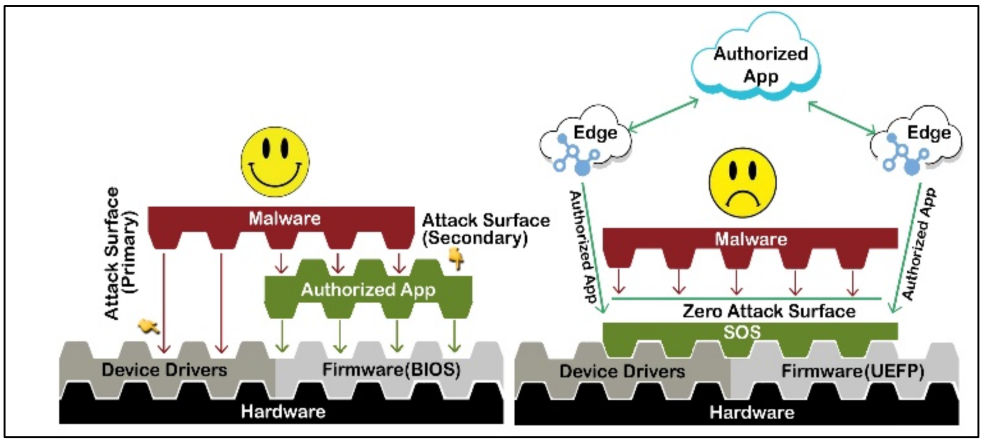


Figure 5. Comparison of architectural constructs of 3rd party permissions and attack surface between legacy IoT systems and ZVC.

In legacy systems, it is normal to expect any software application to create its own attack surface because of the permissions it avails from the OS, firmware, or hardware and inadvertently passes it on as an indirect permission (zero-day vulnerability) to the adversary.

Since ZVC allows only Internet-delivered applications, there is no need for running native applications or a need for third-party permissions that create an attack surface.

A case in point is the infamous Pegasus hack of WhatsApp accounts of some high-profile individuals wherein Pegasus exploited an OS zero-day vulnerability [43]. However, that cannot be the case with an SOS, as the permissions are barred at all levels for all third-party applications, so there is neither a question of zero-day vulnerability, nor any indirect permission being passed on to create a secondary attack surface on the SOS software.

Although originally conceived of as a middleware in the context of mainstream computers (Figure 4b), the SOS was implemented as firmware in the present IoT (hardware wallet) experiments, as illustrated in Figure 5, comparing it with the anatomy of the attack surface in legacy IoT systems. While the residual attack surface cannot be eliminated in the legacy system, there is no residual or secondary attack left after implementing the SOS on a computing device. In absence of any attack surface, no executable program can install or run on the computing device.

Turning on a computer and yet keeping the stored sensitive data offline inaccessible to attackers is another "impossibility" in the prior state-of-the-art that our second ZVC hypothesis challenges by creating ICOS as a tiny USB-mounted NAND Flash-based MUDP 3.0 device, as shown in Figures 4e,f and 6. The device is left mounted on the host computer permanently, but remains offline by default, keeping the sensitive data safe and secure. It is switched on transiently only when executing a transaction, thus minimizing its exposure to network threats. In any case, as explained earlier, the SOS keeps any malware threat at bay (Figure 4e,f) for the tiny fraction of time the device goes online.

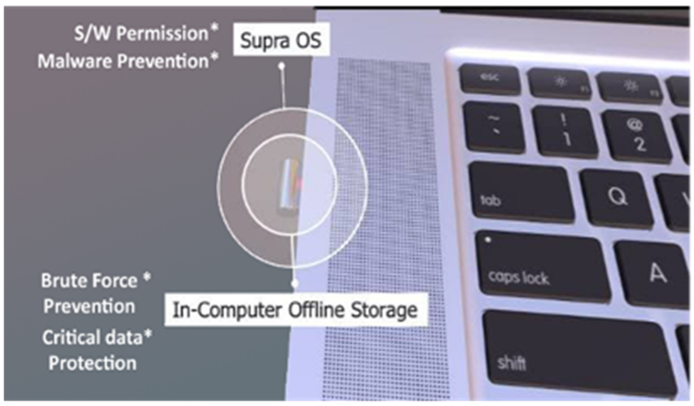


Figure 6. Tiny ZVC device mounted on a USB port of a host PC.

Both hypotheses essentially question the possibility of exploring a new cybersecurity paradigm of zero vulnerability computing. The assumption is that a combination of these two novel encryption-independent security paradigms, when properly designed in specific execution environments (Figure 7) as disclosed in our experiments, may lead to a computing environment with a very high cybersecurity assurance against very strong and capable adversaries. Therefore, the SOS and ICOS are two radically novel components of ZVC, which, when combined with well-crafted encryption mechanisms, such as homomorphic encryption, provide strong security guarantees.

As a proof-of-concept to support and test the ZVC hypotheses, we built a minimalist hardware wallet device to execute secure cryptocurrency transactions. In the simple experiments that we designed to test the efficacy of the SOS and ICOS components, the second hypothesis was easy to test and validate, but the first hypothesis warranted complexities that seemed much more challenging as the SOS hypothesis was originally framed based on mainstream computing systems rather than a minimalist IoT device. With no OS and a very limited computing environment, our original experiment design turned out to be suboptimal, more so now with the benefit of hindsight. However, serendipity has its own way of revealing marvels. The discovery of such a marvel makes these trivial experiments seminal in nature and their interpretation very preliminary, judicious, and guarded.

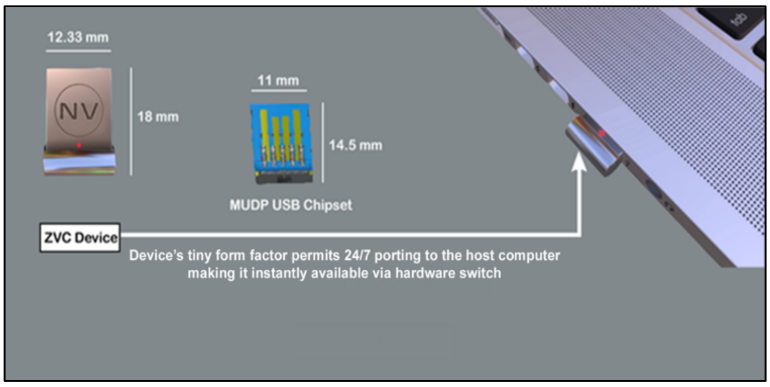

12.33 mm
NV
18 mm
11 mm
14.5 mm
MUDP USB Chipset
ZVC Device
Device's tiny form factor permits 24/7 porting to the host computer making it instantly available via hardware switch

Figure 7. Illustration of the security of a switchable ZVC hardware wallet that incorporates both the SOS and ICOS modules of ZVC.

6. Experiment: Supporting the ZVC Hypotheses

The experiments to support and test the ZVC hypotheses were designed to be conducted remotely at a third-party IMEC laboratory and funded by the European Commission's Horizon 2020 program's NGI Fed4Fire+ cascaded funding (Grant Number 732638). Experiments were conducted at the IMEC's Virtual Wall testbed facility in Ghent, Belgium. These experiments demonstrate the efficacy of the ICOS and SOS modules of ZVC in a highly restricted environment of a USB-mounted, always-connected hardware wallet setting.

6.1. Objectives of This Research

The experiments were designed to support the two hypotheses to achieve zero vulnerability computing. Hence, the following specific objectives were formulated:

(1) Design a ZVC device based on a MUDP 3.0 chipset with a tiny form factor (~17mm length) that facilitates its permanent mounting on a host computer's external USB port without damaging the port.
(2) Build a multi-coin wallet application that supports bitcoins and altcoins as a hardware wallet with cold storage.
(3) Program the ZVC device to completely obliterate its attack surface using the SOS software for Linux.
(4) Test and validate ZVC using a mock malware application that fails to write any data to the ZVC device.
(5) Redesign the MUDP chipset to include a connector on the MUDP PCB that controls power to the chipset.
(6) Redesign the ZVC device housing to include an external on/off switch that controls its online/offline status.
(7) Secure the wallet with fully homomorphic encryption (FHE), partially homomorphic encryption (PHE), and standard encryption to create **cold**, **warm**, and **hot** versions, respectively, and establish transition protocols between them.
(8) Finally, validate the cybersecurity of the ZVC hardware wallet product in a real-world crypto transaction setting.
(9) Disseminate and exploit the results of the experiments.

The goals of the experiments were planned to be achieved in two stages. The first 4 goals were targeted for the Stage 1 experiments and the remaining 5 to be achieved in the Stage 2 experiments. This report presents the results of the Stage 1 experiments.

6.2. Experiment Details

The main aim of this experiment was to support and test the two ZVC hypotheses, which strive to challenge the well-established rules of computer design. This was accomplished by designing, testing, and validating a minimalist prototype that demonstrates the feasibility of ZVC technology in the highly restricted milieu of a USB-mounted hardware wallet device. The objective of this study was also to identify the limitations and areas of improvement that would make replicating the results in mainstream computers possible. To achieve these goals, we utilized an MUDP 3.0 chip built using NAND Flash memory in a metal housing with an LED indicator to show the online/offline status of the device. Figure 8 shows a photograph of the device with its dimensions.

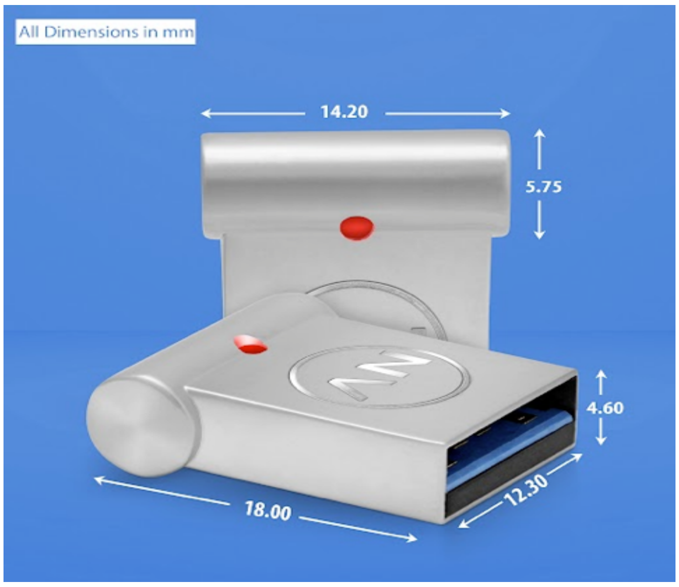


Figure 8. The design and dimensions (mm) of the ZVC powered hardware wallet device.

The device is so tiny that it virtually disappears into the contours of the computer and can therefore remain mounted on the host computer 24/7 without damaging the USB port or the chip. Five units of these devices, along with two units of conventional USB devices as control devices were shipped to a testbed facility in Ghent. The experiments were performed remotely with assistance from the staff of the Virtual Wall 2 testbed of the IMEC labs, who provided the relevant infrastructure with several nodes (Linux computers) for the experiments and also assisted in mounting the devices on the test nodes (interconnected network of Linux computers). The remaining experimental routines were handled remotely. We used the JFed experimental tool to access the nodes remotely and also used the XRDP software to gain remote access to the node (Linux computers) interfaces.

We selected two nodes (Linux computers) for this experiment and marked them as Node0 (target node) and Node1 (hacker node). A ZVC-enabled hardware wallet device and a control device as the standard hardware wallet were mounted on the adjoining USB ports of Node0 or the target node (shown in Figure 9), and the layout of their interfaces accessed using the XRDP software is depicted in Figure 10.

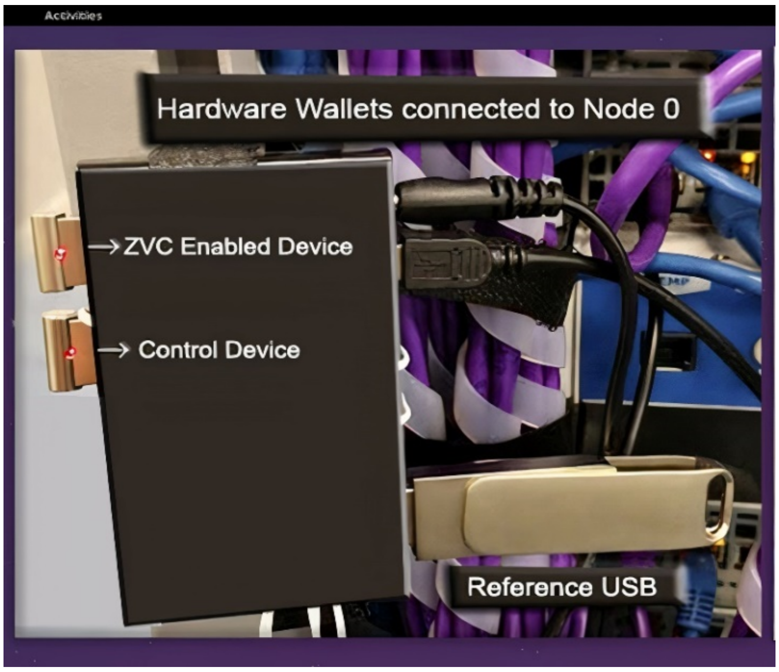
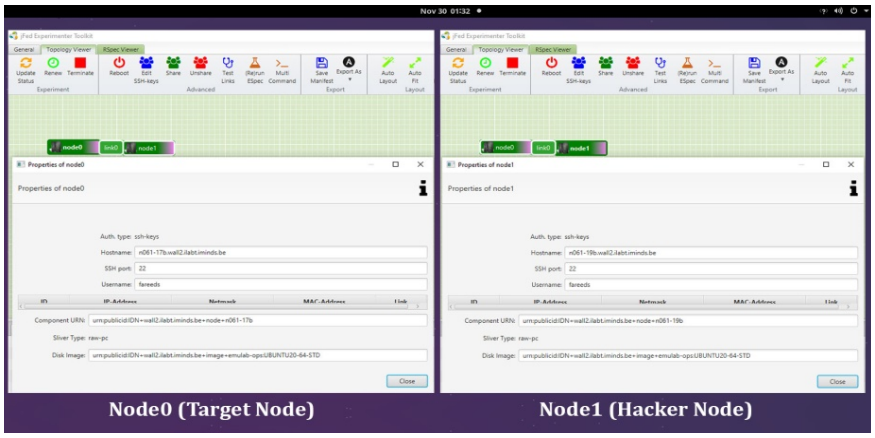


Figure 9. Visual depiction of the test device, the control device, and the reference device mounted on Node0 (target node).

The two selected nodes, Node1 and Node0, were interconnected with the local LAN, and the link between the nodes can be visualized on the interface of the Jfed tool that we used to access these nodes remotely. The following Figure 11 shows the interconnection between the selected nodes.

Node1 was deployed as a hacker node by running a mock malware program designed to remotely infect computers and execute malicious scripts on the victim computer, which in this case is the hardware wallet device mounted on the Node0 computer. Additionally, a conventional USB device was mounted on the third USB port of Node0 as a reference USB device for form factor and operational comparison. The scheme of the mounted devices on Node0 is illustrated in Figure 9. Such a setting simulated a real case scenario where Node0 was a victim computer targeted by the hacker at Node1, wherein both nodes were connected to the Internet. The mock malware running on the hacker Node1 is transmitted to the target victim Node0 and executes its malicious script on all USB devices mounted on Node0 that have unprotected or exposed attack surfaces.



Properties of node0

Auth. type: ssh-keys
Hostname: n061-17b.wall2.ilabt.iminds.be
SSH port: 22
Username: fareeds

ID	IP-Address	Netmask	MAC-Address	Link

Component URN: urn:publicid:IDN+wall2.ilabt.iminds.be+node+n061-17b
Sliver Type: raw-pc
Disk Image: urn:publicid:IDN+wall2.ilabt.iminds.be+image+emulab-ops:UBUNTU20-64-STD

Properties of node1

Auth. type: ssh-keys
Hostname: n061-19b.wall2.ilabt.iminds.be
SSH port: 22
Username: fareeds

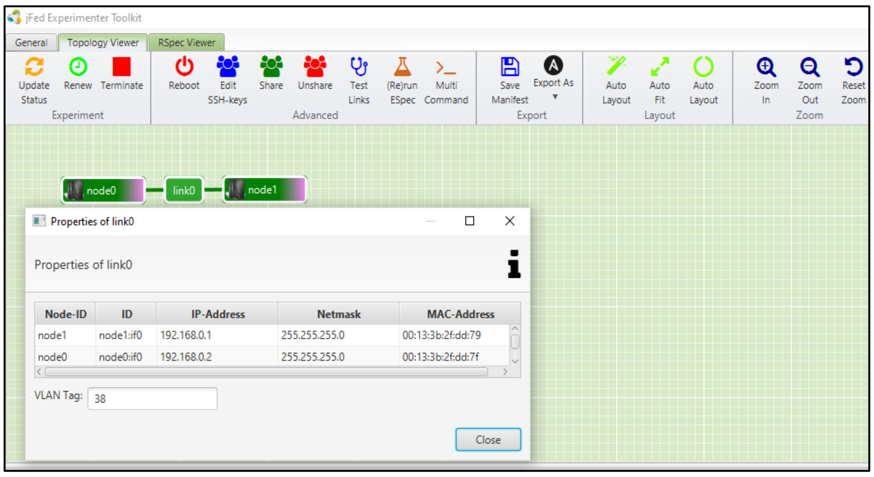
ID	IP-Address	Netmask	MAC-Address	Link

Component URN: urn:publicid:IDN+wall2.ilabt.iminds.be+node+n061-19b
Sliver Type: raw-pc
Disk Image: urn:publicid:IDN+wall2.ilabt.iminds.be+image+emulab-ops:UBUNTU20-64-STD

Figure 10. JFed layout of the user interfaces of the test device and the control device mounted on Node0 (target node).

6.3. Experiment Outline

The selected testbed nodes (Linux computers) were remotely accessed using the JFed tool and XRDP software provided by IMEC's Virtual Wall 2 testbed facilities.



Properties of link0

Node-ID	ID	IP-Address	Netmask	MAC-Address
node1	node1:if0	192.168.0.1	255.255.255.0	00:13:3b:2f:dd:79
node0	node0:if0	192.168.0.2	255.255.255.0	00:13:3b:2f:dd:7f

VLAN Tag: 38

Figure 11. Interconnection between Node0 and Node1.

The staff of the testbed mounted the experimental devices on the USB ports of the selected node. The JFed tool and XRDP software helped us conduct the experiment remotely by implementing and recording events at both experimental nodes. The mock malware was executed on Node1 or the hacker node and was specially designed to detect, infect, and steal data from any device mounted on any of the USB ports of a victim computer targeted for data breach, which in this case was Node0. A mock malware script was designed to copy the .config folder that contains sensitive user information pertaining to the hardware wallet private key of the target user's ETH wallet application running from the hardware wallet device. The following figure (Figure 12) shows the presence of the .config file inside both the ZVC hardware wallet and the control device. The presence of the .config folder inside the ZVC hardware device is highlighted with yellow, while that of the control device is shown with blue highlight.

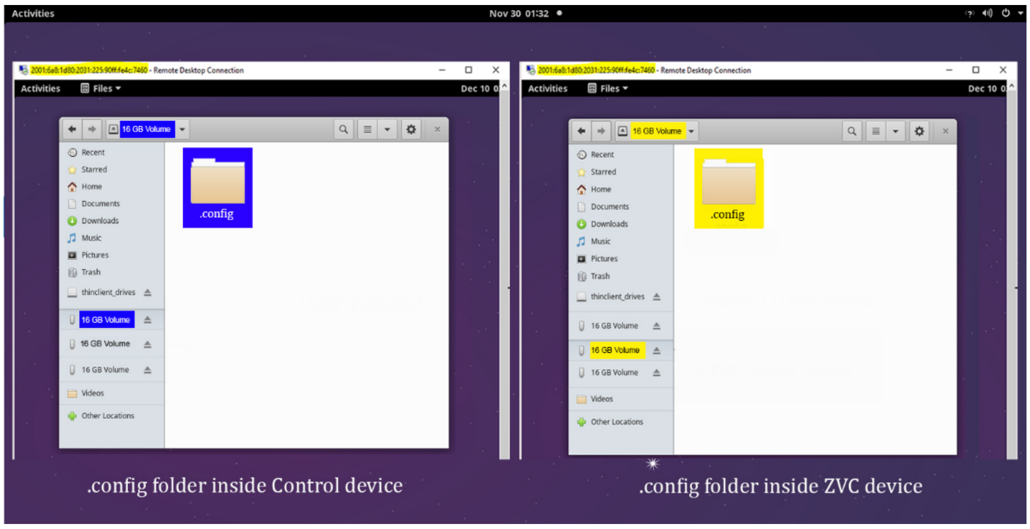
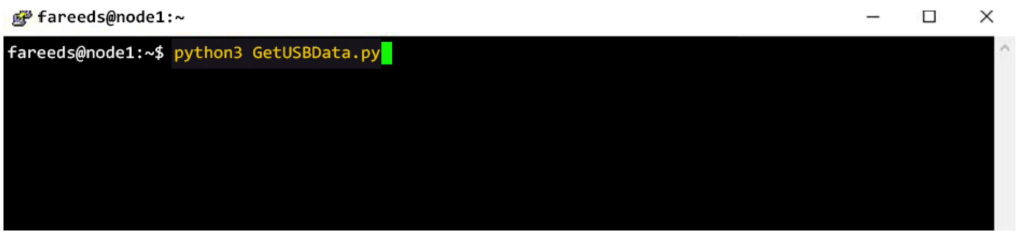


Figure 12. The private key stored in the .config folder in the control (blue) as well as the ZVC (yellow) device.

Access to the hardware ETH wallet is authenticated by the private key saved in the .config folder each time the user accesses the ETH wallet account. Anyone in possession of a private key would thus have access to the target user's ETH wallet. Hence, a hacker's obvious goal would be to steal the private key; therefore, the mock malware was designed to identify hardware wallets mounted on the USB port of the target node and run a script on the victim device to copy the .config folder containing the ETH wallet's private key. Such copied data are sent to Node1 (hacker node). Figure 13 shows the command line interface to transfer the malware from Node1 (hacker node) to Node0 (target node) and get the sensitive data out of the USB device connected to Node0.



```
fareeds@node1:~
fareeds@node1:~$ python3 GetUSBData.py
```

Figure 13. Command line interface with malware command executed for stealing the private keys.

The ZVC-powered hardware wallet with the SOS and ICOS modules (hereinafter referred to as ZVC) was mounted on the first USB port of the target Node0, and another similar hardware wallet without a ZVC program was mounted on Node1, i.e., the target node's second USB port. The former ZVC device served as the test device, and the latter served as a control device. A third USB port on target Node0 had mounted a conventional USB device, which served as a reference device to compare the form factor and operational integrity of the node. The mock malware application was executed on hacker Node1, which identifies target Node0 and infects any hardware device mounted on the USB ports of the target node.

The control device (Case1) and ZVC test device (Case2), mounted on the target nodes, were tested for their hardware wallet integrity using a Ropsten network (also known as "Ethereum Testnet"). Ropsten is an Ethereum test network that allows blockchain development testing before deployment on Mainnet, the main Ethereum network [44]. It is also an ideal test net for recording ERC20 transactions via the ETH wallet, as illustrated in the following scenarios:

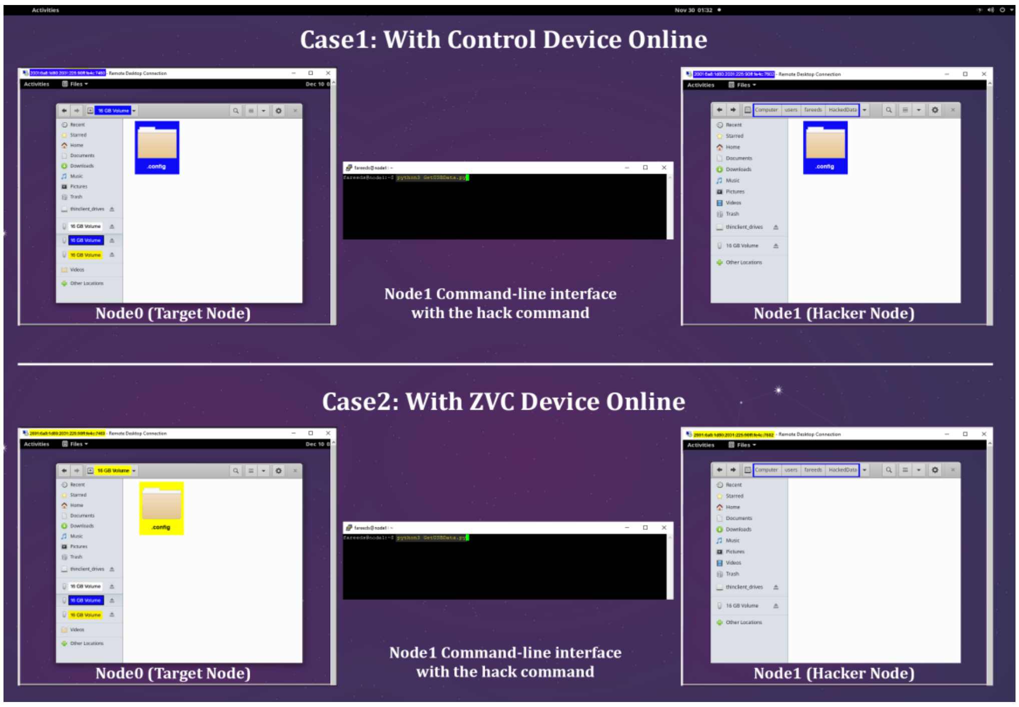
Case1: The permanently mounted control device mounted on the target Node0 has a standard cryptocurrency wallet connected to the Internet 24/7. The wallet application was launched to log in using the private key fetched from the .config folder and perform an ETH transaction on the Ropsten Testnet. Contemporaneously, the malware script ran on the hacker Node1. The activities of both nodes were monitored and recorded to first confirm that the wallet transaction was completed at the target Node0 and, second, the mock malware script at hacker Node1 was running, as well as if the .config folder containing sensitive user information (wallet private keys) was transmitted back to the hacker Node1 from the target Node0.

Case2: In the second experiment, the permanently mounted ZVC device on the target Node0 was switched online only when the wallet application was launched to log in using the private key fetched from the **.config** folder and performed an ETH transaction on the Ropsten Testnet. The ZVC device went offline immediately after the wallet transaction was executed. Contemporaneously, the malware script ran on the hacker Node1. The activities of both nodes were monitored and recorded to first confirm that the wallet transaction was completed at the target Node0; second, we ensured that the mock malware script at the hacker Node1 was running, and third, we checked if the **.config** folder containing sensitive user information (private keys) was transmitted back to the hacker node from the target node. A total of five transactions were recorded with the following binary key performance indicators (KPIs):

1. Data offline while ZVC device is mounted on PC and wallet application not running: Yes/No;
2. Data online while mounted ZVC device runs wallet application: Yes/No;
3. Data always online while control device is mounted on PC: Yes/No;
4. Mock malware able to copy/run malicious code on the control device: Yes/No;
5. Mock malware not able to copy/run malicious code on ZVC device. Yes/No.

6.4. Experiment Result

All five KPIs returned a "Yes" outcome in all five repeats of the Case1 and Case2 implementations of the experiments. In the Case1 implementation, the mock malware script infected the control device on the target node and executed the script on the control device. This was evidenced by the transmission of a copy of the ".config" folder from the target node to the hacker node. This can be visualized in Figure 14, where the UI of both the target node and hacker node shows the presence of the .config folder on both selected nodes (marked in blue). On the other hand, in the Case2 implementation, the ZVC-powered hardware device remained uninfected by the malware script as the SOS software component of ZVC obliterates the attack surface completely by banning all third-party permissions from installing or running any program except the native wallet application. This was observed in the absence of a .config folder (highlighted yellow) on the hacker node UI.


Case1: With Control Device Online
Node0 (Target Node)
Node1 Command-line interface with the hack command
Node1 (Hacker Node)
Case2: With ZVC Device Online
Node0 (Target Node)
Node1 Command-line interface with the hack command
Node1 (Hacker Node)
.config

```
fareeds@node1:~$ python3 GetUSBData.py
```

Figure 14. Screenshots of the command line interface showing the status of the .config file in the control and test devices.

Even after the continuous attempts to infect the ZVC hardware wallet device, the malware failed to find an attack surface to install and run malicious scripts on the ZVC hardware wallet device. Moreover, the ZVC device remained online only during the wallet transaction activity and switched offline despite being mounted on the USB port 24/7. The ability of the malicious program to copy files from the USB location of the target node to the hacker node when the control device was mounted can be attributed to the absence of the ZVC ecosystem, whereas the inability of the same mock malware program to perform any action on the ZVC device connected to the USB port of the same target node can be attributed to the unhackable environment created by the obliteration of the attack surface resulting from the denial of third-party permissions by the ZVC approach.

Thus, our experiments answered both hypothetical questions that we set out to support and test affirmatively.

Hypothesis 1: *Can Supra OS software (SOS) completely obliterate the primary attack surface by denying third-party permissions to all non-native applications?* ***Yes***

Hypothesis 2: *Can a hardware module isolate critical data that require sporadic access to offer* ***in-computer offline storage (ICOS)*** *within the connected device itself without compromising its functionality?* ***Yes***

7. Results' Interpretation and Discussion

Our original disclosure of ZVC described Supra OS (SOS) [45], which obliterated a computer's attack surface, and in-computer offline storage (ICOS) [46] as components of the ZVC framework. Such disclosures were essentially in reference to mainstream computing devices, such as laptops and desktops. However, during our experimentation at IMEC labs with the ZVC hardware wallet device, we had to implement the SOS directly on the NAND Flash firmware, as there was no OS involved. We discovered that such an implementation was more efficient and secure and that ZVC can be directly implemented on non-volatile NAND Flash memory without having to piggyback on any legacy firmware or OS. Because such an approach eliminates all the variables introduced by the complex multilayered legacy computing architecture, it is simpler and easier to control the attack surface of IoT devices with limited processing power.

Complexities and a multitude of variables create a situation similar to the pre-solid-state electronics era, wherein moving parts in electronic devices of that era made the devices fragile, bulky, energy inefficient, and vulnerable to external risks. The advent of solid-state technology in the 1960s-1970s revolutionized electronics and made portable computing devices possible today. Although some device manufacturers tend to market their computing devices as solid-state simply because their devices are built using one or more of their components using solid-state hardware components, they cannot really be called solid-state, as these devices run multi-layered legacy software systems. The multi-layered legacy software system running on any device introduces variabilities and, consequently, vulnerabilities to the devices. The more complexities and variabilities present in a software system, the more vulnerable it becomes. The layered architecture of traditional computers introduces complexities and variability. The variabilities in the layered architecture of legacy software systems are analogous to those in the moving parts of electronic devices. Because ZVC essentially bans all third-party permissions at each layer of the legacy computers, thereby merging all the layers between the hardware and human–computer interface (HCI), it creates a scenario akin to no moving parts of solid-state electronics. Hence, the direct implementation of ZVC on the nonvolatile memory chip of a computing device eliminates the need for piggybacking on the firmware of the device or the OS. This can potentially lead to the conceptualization of a radically novel concept of solid-state software-on-a-chip (3SoC) with no conventional distinguishable layers of firmware, drivers, OS, and application (all moving parts) between the hardware and the HCI. This 3SoC approach creates a compact solid-state software milieu that is ideal for IoT devices with limited resources. Moreover, because IoT devices are mostly single-purpose devices with little or no need for third-party applications to run on them, banning all third-party permissions would not adversely affect their usability.

As illustrated in Figure 15, the multilayered legacy computing system is built on granting third-party permissions at each layer, thus allowing an attack surface at each layer to simulate a situation analogous to the moving parts scenario of old-generation electronic devices. Thus, the proposed ZVC ecosystem can merge all the layered components in the legacy software by banning all third-party permissions, thus obliterating the attack surface and rendering it free of vulnerabilities.

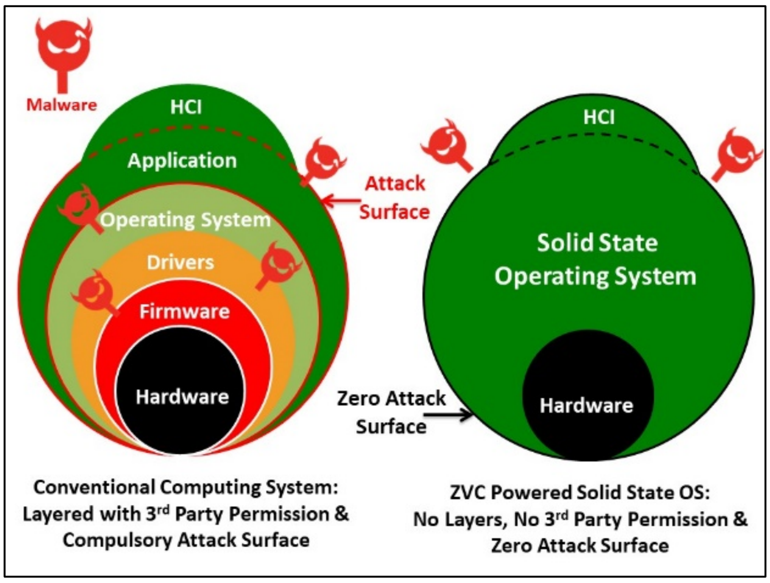

Malware
HCI
Application
Operating System
Drivers
Firmware
Hardware
Attack Surface
Zero Attack Surface
HCI
Solid State Operating System
Hardware
Conventional Computing System: Layered with 3rd Party Permission & Compulsory Attack Surface
ZVC Powered Solid State OS: No Layers, No 3rd Party Permission & Zero Attack Surface

Figure 15. Diagrammatic illustration of the conventional multi-layered computer architecture in comparison to the projected compact zero attack surface design of ZVC.

Although our ZVC hardware wallet experiments supported and tested the two ZVC hypotheses and met the objectives of the study, the results must be interpreted with caution, warranting further rigorous validation of the concept beyond the limited environment of these experiments. If we had the advantage of hindsight, our experimental design, which revolved around the mainstream computing architecture, would have been more relevant to the tested hardware wallet IoT device. In our current experiments, the ZVC software was installed on the NAND chip firmware with the following caveats:

First, the obliteration of the attack, although adequate for the device tested, may not be complete, as the hardware supplier's firmware may remain a source of third-party permissions, and therefore vulnerable.

Second, the restricted environment and limited processing power of a hardware wallet IoT device limit the extrapolation of the results to more complex mainstream computers.

Third, although zeroing of the attack surface and maintaining sensitive data offline in a connected device may address most cyber-attacks, this cannot rule out the possibility of side-channel attacks if the connected device remains physically accessible to a hardware hacker and remains operational during the attack.

Fourthly, although the experimental results led to the formulation of two new hypotheses (Section 6), the device we developed technically neither qualifies as quantum-resistant, nor as solid-state software-on-a-chip (3SoC) as the 3SoC hypothesis we formulated for future research defined in Section 9.

Finally, the results of this research helped us fine-tune the definition of ZVC:

ZVC is a cybersecurity paradigm that proposes a new zero attack surface computer architecture that restricts all third-party applications exclusively to a web interface only, declining permissions for any utilization of computing resources by any non-native program and creates a switchable in-computer offline storage for securing sensitive data at the user's behest.

8. Work Limitations

With the advantage of hindsight, we realize that our experiment design was not as rigorous and meticulous as such a far-reaching concept deserves. This is essentially because our original approach was designed as the Supra OS (SOS), which obliterates the attack surface of a computer's operating system to provide one more tool to the cybersecurity arsenal. We had absolutely no apprehension that the concept would evolve to challenge the very architecture of computing systems and propose a radical paradigm shift in designing future computers. The fact that all legacy hardware and all software were inherently designed to grant third-party permissions was not on our radar screen when we designed the experiments. Third-party permissions can be exploited at the hardware and firmware level besides the OS and the application layer. Broadening the scope of the SOS beyond the OS had sweeping consequences. Moreover, the experiments were designed to support the hypothesis and not to prove the hypothesis. For these reasons, therefore, the results should be interpreted with great care and caution. More rigorous validation is warranted before extrapolating the results in real-world scenarios.

9. Future Prospects and New Hypotheses

ZVC is a radically novel concept that needs to be extensively researched to explore its possible implementation beyond the limited use of a hardware wallet. Our experiments tested the technological feasibility of the ICOS and SOS modules of the ZVC in a highly restricted environment of a USB-mounted always-connected device setting. The ICOS module keeps the data secure offline, except when required for processing. The SOS module completely obliterates the attack surface on the hardware wallet by banning third-party permissions. While these findings are preliminary, demonstrating the potential feasibility of the ZVC concept, further validation in more challenging mainstream computing environments is urgently required. Moreover, the results of this study also suggest exploring the ZVC approach in two distinct areas of concern in computer research, which can be defined by formulating two new hypotheses as follows:

1. *Because ZVC security is encryption-independent, will it be quantum-resistant by design?*
2. *As the ZVC architecture lacks layering, rendering it conceptually analogous to the zero moving parts nature of solid-state electronics, will it deliver the same advantages to computers as solid-state electronics did to revolutionize the electronics industry in the 1960s–1970s?*

Given the robustness, energy efficiency, portability, and resilience that solid-state electronics introduce to any product, there is an urgent need to test and validate these hypotheses. If the 3SoC concept is established, it has the potential to emulate the same level of revolutionary success as the solid-state concept achieved in the electronics industry, which made modern-day gadgets possible. Testing and validating these hypotheses will have an unfathomable impact on the design of future computers and will revolutionize cybersecurity.

Author Contributions: Conceptualization, F.R.; Formal analysis, T.B.; Funding acquisition, P.V.D.; Investigation, F.R.; Methodology, F.R. and T.B.; Project administration, B.V. and P.V.D.; Resources, B.V.; Software, T.B.; Supervision, B.V. and P.V.D.; Validation, B.V. and P.V.D.; Writing—original draft, T.B.; Writing—review and editing, F.R. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the European Commission's Horizon 2020 program's NGI Fed4Fire+ cascaded funding under Grant # 732638.

Data Availability Statement: Not applicable, the study does not report any data.

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

References

1. Morgan, S. Cybercrime to Cost the World $10.5 Trillion Annually by 2025. Available online: https://cybersecurityventures.com/cybercrime-damages-6-trillion-by-2021/ (accessed on 13 November 2020).
2. Satoh, A.; Fukuda, Y.; Hayashi, T.; Kitagata, G. A Superficial Analysis Approach for Identifying Malicious Domain Names Generated by DGA Malware. *IEEE Open J. Commun. Soc.* **2020**, *1*, 1837–1849. [CrossRef]
3. Adhikary, T.; Jana, A.D.; Chakrabarty, A.; Jana, S.K. The Internet of Things (IoT) Augmentation in Healthcare: An Application Analytics. In *International Conference on Intelligent Computing and Communication Technologies*; Springer: Singapore, 2019; pp. 576–583. [CrossRef]

4. Davis, G. 2020: Life with 50 billion connected devices. In Proceedings of the 2018 IEEE International Conference on Innovative Research and Development (ICCE), Las Vegas, NV, USA, 12–15 January 2018; p. 1. [CrossRef]
5. Greco, G.; Montinaro, N. The phenomenon of cybercrime: From the transnational connotation to the need for globalization of justice. *Eur. J. Soc. Sci. Stud.* **2020**, *6*. Available online: https://doi.org/10.46827/ejsss.v6i1.956 (accessed on 27 June 2022). [CrossRef]
6. Manjoo, F. A Future Where Everything Becomes a Computer Is as Creepy as You Feared. New York Times. Available online: https://www.nytimes.com/2018/10/10/technology/future-internet-of-things.html (accessed on 10 October 2018).
7. Aftergood, S. Governments want your smart devices to have stupid security flaws. *Nature* **2018**, *560*, 550–551. [CrossRef]
8. Woodford, C. A Brief History of Computers. Available online: http://nmis.isti.cnr.it/casarosa/BDG/supporto/A%20brief%20history%20of%20computers.pdf (accessed on 27 June 2022).
9. Cao, K.; Liu, Y.; Meng, G.; Sun, Q. An overview on edge computing research. *IEEE Access* **2020**, *8*, 85714–85728. [CrossRef]
10. Zhang, X.; Chen, H.; Zhao, Y.; Ma, Z.; Xu, Y.; Huang, H.; Yin, H.; Wu, D.O. Improving cloud gaming experience through mobile edge computing. *IEEE Wirel. Commun.* **2019**, *6*, 178–183. [CrossRef]
11. Bartel, A.; Klein, J.; Le Traon, Y.; Monperrus, M. Automatically securing permission-based software by reducing the attack surface: An application to android. In Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering, Essen, Germany, 3–7 September 2012; pp. 274–277.
12. Soni, R. SSL, Security, and Authentication. In *Nginx*; Apress: Berkeley, CA, USA, 2016; pp. 195–208.
13. Weber, S.; Kaufman, D.; Thomas, D.; Cohn, A. *Cybersecurity Futures 2025 Insights and Findings*; Center for Long–Term Cybersecurity, University of Berkley: Berkeley, CA, USA, 2019; pp. 10–18.
14. Anoop, S.; Ou, X. Security risk analysis of enterprise networks using probabilistic attack graphs. In *Network Security Metrics*; Springer: Cham, Switzerland, 2017; pp. 53–73.
15. National Vulnerability Database. Available online: https://nvd.nist.gov/vuln-metrics/cvss (accessed on 27 June 2022).
16. CVE Details. Available online: https://www.cvedetails.com/ (accessed on 27 June 2022).
17. DVE Details. Available online: https://www.cvedetails.com/top-50-products.php (accessed on 27 June 2022).
18. Statista [Online] Internet of Things (IoT) Connected Devices from 2015 to 2025 (in Billions). Available online: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/ (accessed on 27 June 2022).
19. Dickson, B. What Bruce Schneier Teaches Us about IoT and Cybersecurity. Techtalk. Available online: https://bdtechtalks.com/2016/11/29/what-bruce-schneier-teaches-us-about-iot-and-cybersecurity/ (accessed on 29 November 2016).
20. The AV Test Institute GmbH. Malware Statistics. Available online: https://www.av-test.org/en/statistics/malware/ (accessed on 10 March 2022).
21. Canella, C.; van Bulck, J.; Schwarz, M.; Lipp, M.; von Berg, B.; Ortner, P.; Piessens, F.; Evtyushkin, D.; Gruss, D. A systematic evaluation of transient execution attacks and defenses. In Proceedings of the 28th USENIX Security Symposium, Santa Clara, CA, USA, 14–15 August 2018; pp. 249–266.
22. Filho, A.S.; Rodríguez, R.J.; Feitosa, E.L. Reducing the Attack Surface of Dynamic Binary Instrumentation Frameworks. In *Developments and Advances in Defense and Security. Smart Innovation, Systems and Technologies*; Rocha, Á., Pereira, R., Eds.; Springer: Singapore, 2020; Volume 152, pp. 3–13.
23. Stafford, V.A. *Zero Trust Architecture*; NIST Special Publication: Gaithersburg, MD, USA, 2017; pp. 207–800.
24. Manadhata, P.K.; Wing, J. An attack surface metric. *IEEE Trans. Softw. Eng.* **2010**, *37*, 371–386. [CrossRef]
25. Tung, Y.-H.; Chuang, C.-J.; Shan, H.-L. A framework of code reuse in open source software. In Proceedings of the 16th Asia-Pacific Network Operations and Management Symposium, Hsinchu, Taiwan, 17–19 September 2014; pp. 1–6. [CrossRef]
26. Stellios, I.; Kotzanikolaou, P.; Grigoriadis, C. Assessing IoT Enabled Cyber-Physical Attack Paths Against Critical Systems. *Comput. Secur.* **2021**, *107*, 102316. [CrossRef]
27. Zhang, M.; Xin, Y.; Wang, L.; Jajodia, S.; Singhal, A. CASFinder: Detecting Common Attack Surface. In *Data and Applications Security and Privacy XXXIII*; Foley, S., Ed.; DBSec, Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2019; Volume 11559, pp. 338–358.
28. Cran, J. The New Application Attack Surface. Kenna Security. Available online: https://www.kennasecurity.com/blog/the-new-application-attack-surface/ (accessed on 23 May 2019).
29. Cybersecurity Ventures, Herjavec Group. Hypercopolypse: A Cybercrime Revelation. Cyentia Cybersecurity Research Library. Available online: https://library.cyentia.com/report/report_001392.html (accessed on 12 October 2018).
30. Castelvecchi, D. The race to save the Internet from quantum hackers. *Nature* **2022**, *602*, 198–201. [CrossRef] [PubMed]
31. Rozell, D.J. Cash is king. *Nature* **2022**, *16*. [CrossRef]
32. Grimes, R.A. *Cryptography Apocalypse: Preparing for the Day When Quantum Computing Breaks Today's Crypto*; John Wiley & Sons: Hoboken, NJ, USA, 2019.
33. Zhang, Q.; Liu, L.; Pu, C.; Dou, Q.; Wu, L.; Zhou, W. A Comparative Study of Containers and Virtual Machines in Big Data Environment. In Proceedings of the IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, USA, 2–7 July 2018; pp. 178–185. [CrossRef]
34. Zarca, A.M.; Bernabe, J.B.; Farris, I.; Khettab, Y.; Taleb, T.; Skarmeta, A. Enhancing IoT security through network softwarization and virtual security appliances. *Int. J. Netw. Manag.* **2018**, *28*, e2038. Available online: https://doi.org/10.1002/nem.2038 (accessed on 27 June 2022).

35. Apostolopoulos, T.; Katos, V.; Choo, K.K.R.; Patsakis, C. Resurrecting anti-virtualization and anti-debugging: Unhooking your hooks. *Future Gener. Comput. Syst.* **2021**, *116*, 393–405. [CrossRef]
36. Gupta, U. Comparison between security majors in virtual machine and linux containers. *arXiv* **2015**, arXiv:1507.07816.
37. MacAskill, E.; Thielman, S.; Oltermann, P. WikiLeaks Publishes 'Biggest Ever Leak of Secret CIA Documents'. The Guardian, 2017. Available online: https://www.theguardian.com/media/2017/mar/07/wikileaks-publishes-biggest-ever-leakof-secret-cia-documents-hackingsurveillance (accessed on 27 June 2022).
38. Guri, M.; Monitz, M.; Elovici, Y. Bridging the air gap between isolated networks and mobile phones in a practical cyber-attack. *ACM Trans. Intell. Syst. Technol.* **2017**, *8*, 50. [CrossRef]
39. Johnson, D.B. The Defense Advanced Research Projects Agency I Looking for Security That Can Better Track and Protect Sensitive Data as It Moves from Highly Secure Systems to Insecure Ones. Available online: https://defensesystems.com/2019/01/darpa-looks-to-guard-the-air-gap/192351/ (accessed on 8 January 2019).
40. Guri, M.; Zadov, B.; Elovici, Y. ODINI: Escaping Sensitive Data from Faraday-Caged, Air-Gapped Computers via Magnetic Fields. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 1190–1203. [CrossRef]
41. Fazal, R. A Novel Method and System for Using Optical Disk Drive as Biometric Card Reader for Secure Online User Authentication. U.S. Patent No. 7,228,424, 5 June 2007. Available online: https://patents.google.com/patent/US7228424 (accessed on 27 June 2022).
42. Younis, A.A.; Malaiya, Y.K. Relationship between Attack Surface and Vulnerability Density: A Case Study on Apache HTTP Server. In Proceedings of the International Conference on Internet Computing (ICOMP), Las Vegas, NV, USA, 16–19 July 2012; pp. 1–7.
43. Pegg, D.; Cutler, S. What Is Pegasus and How Does It Hack Phones? The Guardian. Available online: https://www.theguardian.com/news/2021/jul/18/what-is-pegasus-spyware-and-how-does-it-hack-phones (accessed on 18 July 2021).
44. Zhang, L.; Lee, B.; Ye, Y.; Qiao, Y. Ethereum transaction performance evaluation using test-nets. In *European Conference on Parallel Processing*; Springer: Cham, Switzerland, 2019; pp. 179–190.
45. Raheman, F. A Novel Supra Operating System Software for The Next Generation Web 3.0. U.S. Patent Application 63/202,188, 31 May 2021.
46. Raheman, F. In-Computer Offline Storage (ICOS) to Achieve Zero Vulnerability Computing (ZVC). U.S. Patent Application 63/228,122, 1 August 2021.