electronics

MDPI

Article

Vulnerabilities' Assessment and Mitigation Strategies for the Small Linux Server, Onion Omega2

Darshana Upadhyay [1], Srinivas Sampalli [1,*] and Bernard Plourde [2]

1 Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 4R2, Canada; darshana@dal.ca
2 Research & Development Department, Technology Sanstream, Ottawa, ON K2R 1C1, Canada; bplourde@sanstream.ca
* Correspondence: srini@cs.dal.ca

Received: 8 May 2020; Accepted: 5 June 2020; Published: 10 June 2020

check for updates

Abstract: The merger of SCADA (supervisory control and data acquisition) and IoTs (internet of things) technologies allows end-users to monitor and control industrial components remotely. However, this transformation opens up a new set of attack vectors and unpredicted vulnerabilities in SCADA/IoT field devices. Proper identification, assessment, and verification of each SCADA/IoT component through advanced scanning and penetration testing tools in the early stage is a crucial step in risk assessment. The Omega2, a small Linux server from Onion™, is used to develop various SCADA/IoT systems and is a key component of nano power grid systems. In this paper, we report product level vulnerabilities of Onion Omega2 that we have uncovered using advanced vulnerability scanning tools. Through this research, we would like to assist vendors, asset owners, network administrators, and security professionals by creating an awareness of the vulnerabilities of Onion Omega2 and by suggesting effective mitigations and security best practices.

Keywords: SCADA; IoT; embedded devices; vulnerability assessment; webserver; Onion Omega2

1. Introduction

There has been a surge in the deployment of internet of things (IoT) with supervisory control and data acquisition (SCADA) systems to control industrial infrastructures across open access networks. While this has provided better control and manageability, it has also exposed such systems to cyber threats [1]. Devices in IoT/SCADA are equipped with microcontrollers for processing information. However, these microcontrollers have limited computational power and bandwidth [1]. This makes the deployment of sophisticated security features infeasible. Recently, the internet engineering task force (IETF) has released RFC 8576 that detects embedded device vulnerabilities such as object cloning, vulnerable software, malicious substitution, denial of service, and firmware attacks as the main threats for IoT/SCADA-based systems [2]. Generally, these devices are vulnerable to cyber attacks owing to weak structural design and bad coding practices during the development life cycle which may expose the entire control system to the outside world [3]. Hence, secure design and development of SCADA products is the primary step towards securing SCADA systems. One example of a bad programming practice is the vulnerability introduced in a programable logic controller (PLC) product owing to hardcoded username and password running in the WinCC database, PCS SCADA software [4]. This became an entry point for the Stuxnet worm which did substantial damage to the Iran nuclear plant. Also, the Kudankulam nuclear power plant (KKNPP) in India, was recently hacked in 2019 using malware attacks [5]. The Dtrack tool was used to gather the information of entire networks and their devices for potential vulnerabilities for exploitations [5].

The common loopholes in SCADA products include buffer overflow, lack of bounds checking, command injections, cross-site scripting and directory path traversal [6]. Most of these SCADA vulnerabilities of the top vendor products were revealed in the SCADA vulnerabilities and exposures database [7]. To overcome these weaknesses, vulnerability assessment of the specific product is one of the crucial factors while working on IoT/SCADA based critical applications. We need to manage the vulnerability assessment and risk management by evaluating, prioritizing and addressing the most immediate challenges by testing and addressing vulnerabilities on an ongoing basis as risk includes business disruption, financial loss and sometimes even loss of life [8].

We have considered a Solar-PV nano-grid power system as a case-study to assess and enhance the security of SCADA systems. This system consists of various field components comprised of solar panels, smart junction boxes and ZigBee coordinators. The Omega2, which is a small embedded Linux server by Onion™, manages these field devices by providing instructions according to the set points for the normal and critical events. It has a 580 MHz MIPS CPU, 128 MB memory, and 32 MB storage along with built in Wi-Fi, microSD slot and Linux operating system [9]. Figure 1 shows the Onion Omega2 IoT device [9]. Furthermore, in a traditional IT environment, clients (system-owners and technicians) control the field equipment through Onion Omega2 from their mobile devices. The Onion Omega2 is also known as the master terminal unit (MTU) or SCADA master and serves as a central controller and acts as a heart of the system.

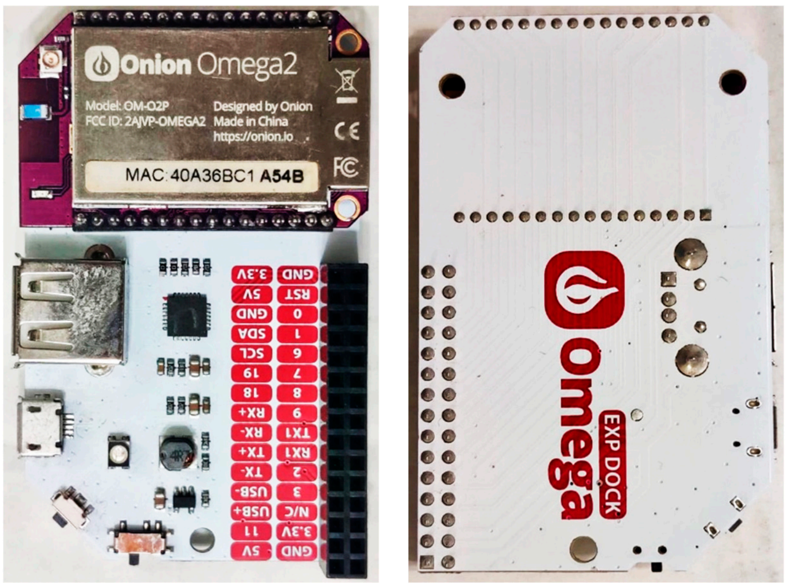


Figure 1. Omega2 hardware [9].

The product level vulnerability assessments are becoming more challenging for vendors owing to lack of knowledge regarding the implementation of cyber security standards in embedded system devices [10]. These evaluations are mostly left to test engineers without giving proper device specifications and access to the source of the firmware source. Moreover, it becomes more challenging owing to specialized, custom, real-time, and limited interface options [11]. In order to debug such product level vulnerabilities, specialized tools are required. Many standard tools are recommended; however, those are often unavailable due to the high cost of software licenses [12]. Hence, there is a high probability of the existence of such vulnerable devices. The evidence of such weaknesses in top vendor products has been revealed in the SCADA vulnerabilities and exposures database [7]. Furthermore, the vendor of Onion Omega2 has mentioned a few known firmware related issues in [13] and has been working for the product improvement.

This paper primarily focuses on the vulnerabilities of SCADA embedded systems at the device level, by examining and testing Onion Omega2 firmware. The paper highlights various product level vulnerabilities of Onion Omega2 where few fixes are applicable at the network level and few required vendor's patches are available. Our research can help owners, vendors, and researchers to leverage the security of a SCADA system by identifying the vulnerabilities in Onion Omega2. The aim of our contribution is to spread awareness with regards to product level weaknesses over the SCADA landscape to help in the secure implementation and deployment of SCADA products in automated industries.

The paper is organized as follows. Section 2 discusses related work based on product level vulnerabilities followed by effectiveness of scanning and testing tools and the vulnerability management lifecycle. Section 3 focuses on the experimental setup of the Solar-PV nano-grid testbed impacted with Onion Omega2. Section 4 emphasizes the various product level vulnerabilities of Onion Omega2 and related mitigation techniques. Concluding remarks are presented in Section 5.

2. Background

To protect industrial control systems from malicious activities, various ongoing research and vulnerability assessment methodologies have been adopted. Several open-source, standard scanning and penetration tools are being used to locate unpatched devices or other weak points in the networks [14]. The proper awareness and effective utilization of such scanning tools can significantly improve the assessment process. Well-known, open-source information-gathering tools such as Nmap, Nikto and Sparta are used to identify open ports and services [15]. Furthermore, Nessus, OpenVAS, FoundScan and Internet Security Scanner have been used as popular scanning tools. These tools allow us to scan network devices and check them against their databases containing thousands of records for known vulnerabilities. The open-source OpenVAS vulnerability scanner developed by Greenbone Security is used to test various protocols and networks. Many vulnerability scanners and penetration testing tools are available in the Kali-Linux operating system [16]. Moreover, various test tools have been proposed and implemented in academic research to determine network flaws based on grammar and fuzzy logic methodology [17]. The PROTOS project group has developed tools using syntax-based generation according to the protocol type [18]. A popular search engine, Shodan, contains information of more than 600 million publicly available IoT devices such as various ports information and banner data information, etc. This information is used to assess the weaknesses of SCADA/IoT devices in an attempt to mitigate the attacks [19].

Vulnerability management constantly progresses through a lifecycle that includes setting a baseline which indicates the starting point used for comparative analysis of the given products, assessment of vulnerabilities, risk assessment, remediation, verification, and monitoring (Figure 2). Furthermore, security evaluation of the embedded systems focuses on firmware analysis owing to loopholes in firmware design and development lifecycle. At present, various models and categories of the embedded products are launched by different manufacturers most often, however, the decoding solution is not necessarily applied to each and every version of these products [20]. These bring product level vulnerabilities in such devices, tools such as Binwalk, Firmware Reverse Analysis Console (FRAK), Interactive Disassembler (IDA) and Binary Analysis Toolkit (BAT) are used to decompress the filesystem of firmware [21]. The Binwalk tool is used to decompose the binary file and extract the metadata from it [22]. FRAK evaluates the data provided by the equipment service provider and decompress it. However, Binwalk has more capability to decompress the file compared to FRAK. BAT uses GPLtool, which recursively extracts the files from the binary firmware and also provides support to segment the document [20].

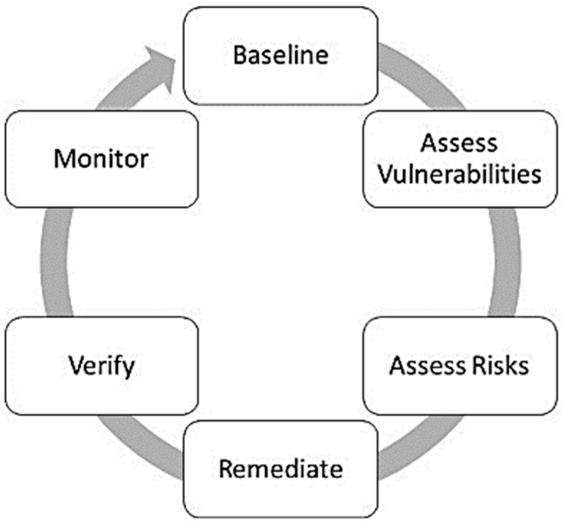


Figure 2. Vulnerability management life cycle.

Various frameworks have been proposed by researchers for life cycle assessments of infrastructure. Creery and Byres [23] propose a complete security assessment process model for the evaluation of control systems. To protect industrial infrastructure, I-VAM (infrastructure vulnerability assessment model) has been proposed for medium-sized clean water systems [24]. The control system cyber security self-assessment tool (CS2SAT) was developed by the Idaho National Laboratory to evaluate control system security [25]. This tool has the capability to systematically evaluate the product by collecting all the necessary information from various resources to identify flaws in the system. Sandia National Laboratories has developed the information design assurance red team (IDART) to evaluate the security strength of SCADA systems [26]. Most of these assessment techniques have been proposed for identifying security flaws in control and monitoring systems in general, with limited application to SCADA systems. Furthermore, the previous approaches are mainly system-level assessment tools rather than at the device-level. In order to identify all security issues in SCADA systems, vulnerability assessment must be done for each component. The approach should begin with risk assessment followed by vulnerability evaluation to validate the security of embedded devices. After a successful assessment, corresponding mitigation strategies should be applied to address each vulnerability.

In the area of risk assessment, studies in [27–29], focused on risks to IoT systems in general, which covers three major aspects of the IoT architecture, namely, behavioural, physical, and operational facets. In [30], the authors propose the risk assessment techniques based on the Allegro methodology to identify cyber and physical security vulnerabilities of IoT-based systems. The study emphasizes vulnerability assessment of five system components, namely, sensors, cloud servers, gateways, smartphone apps, and application programming interfaces. In [29], a phone-out-only policy and virtual environment strategies were proposed to ensure the security of remote monitoring and control systems. These approaches have mainly focused on either cyber or physical aspects of the risk assessment. There has been no significant work that has specifically focused on IoT/SCADA product level vulnerability assessments.

In this paper, we demonstrate a step-by-step process to evaluate product level vulnerabilities of Onion Omega2, which focuses on both vendor and network-level fixes. Moreover, this assessment strategy not only adheres to the best practices, but also provides the roadmap to build and assess other secure embedded devices which include micro-controllers, platforms, and customizable operating systems.

We started our assessment process with basic port scans followed by webserver and DNS server assessment. Through this, we have evaluated common loopholes in SCADA products, which include buffer overflow, lack of bounds checking, command injections, cross-site scripting, and directory path traversal. We then assessed the security strength of remote login and Wi-Fi communication protocols.

Furthermore, SSL/TLS (secure socket layer / transport layer security) security configuration is evaluated to analyze the strength of security protocols. By using a similar systematic vulnerability assessment process, security practitioners can determine the security weaknesses of any embedded devices from the primary to the advanced level.

3. Onion Omega2 IoT Device

This section briefly explains the experimental setup of a Solar-PV nano-grid system to showcase the role of Onion Omega2 in the proposed testbed. At a high level, the Solar-PV nano-grid system is divided into three operational zones as shown in Figure 3. The first zone consists of the actual assets—the solar panel, smart junction box (SJB) and Zig-bee coordinators. Smart junction boxes continuously coordinate with Zigbee-coordinators to monitor and control the solar panels to manage flow of energy according to the instructions received from the MTU. The second zone is entirely dependent on the gateway server in which Onion Omega2 acts as a purpose-built SCADA equipment called the SCADA master (or master terminal unit). This server manages field-devices. It monitors and controls the system for normal events as well as for the critical issues such as the energy set point being exceeded by a malfunctioning system. The communication between Zone1 and Zone2 is established using UART (Universal Asynchronous Receiver/Transmitter), which transmits and receives serial data. The third zone is the traditional IT environment—where clients (operators and/or technicians) can control the field equipment through SCADA using the MTU from a mobile device. The communication is established amongst mobile users and the SCADA master using the WPA2 (Wi-Fi protected access 2) Wi-Fi protocol. Collectively, these zones are highly complex, often distributed and well connected.

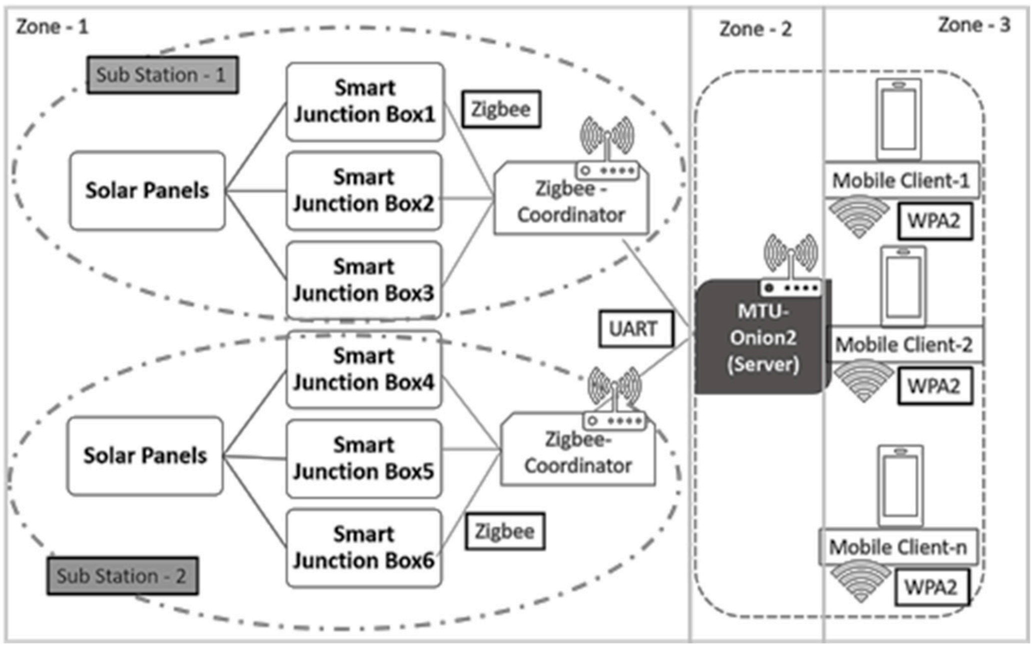


Figure 3. Block diagram of the Solar-PV nano-grid system.

As with any complex system, accidents happen, and errors occur due to manual operations and inadequate training. Additionally, such systems are high-value targets for attackers which may take down the electric grid, communication services and/or emergency services by identifying loopholes in the system. For that, backdoors of vulnerable products are often used [31]. Hence, one of the critical requirements of Nano PV systems is to deploy secure products.

Proper awareness of unpatched vulnerabilities and zero-day attacks is the key to risk assessment. Therefore, our primary task is to identify loopholes in the proposed testbed by assessing each device through various scanners and penetration tools. This approach will help us identify potential breaches in the system which will lead us to adopt proper prevention mechanisms and enhance the security of power grids.

4. Vulnerability Assessments of Onion Omega2

Vulnerability assessment is typically a highly subjective process; it requires powerful analytical strategy and computational methodology [10]. For a thorough review of vulnerability assessment of the Onion Omega2, we have followed standard vulnerability assessment tools and techniques. We started the security assessment process of Onion Omega2 by conducting firmware analysis using the Binwalk tool. Further, we have used standard tools such as Nikto, Sparta, OpenVAS and Nessus to analyze the scan results of Onion Omega2 starting from basic port scans to advance level testing. We describe the scan results and mitigation techniques of Onion Omega2 in the following section. The section is mainly divided into two parts according to remediation strategies:

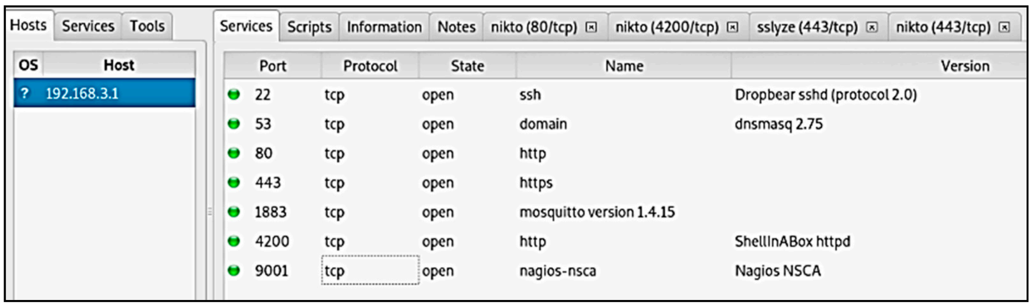
1. Vendor level fixes
2. Network/Administrator level fixes

4.1. Vendor Level Fixes

The following section highlights the vulnerability assessment of the Onion Omega2 which focuses on vendor level fixes. We provide a comprehensive summary of vendor level fixes in Table 1 (which appears later in this section).

4.1.1. Open Ports/Services

Through port scanning we can identify which ports and services are open and analyze how they respond to the queries. This is a popular method for gathering information about the target. This information may be used as a precursor to an attack, hence testing the system using Netstat or Nmap is always a primary step to secure the system by identifying unnecessary open ports that should be blocked. Therefore, we scanned all the ports and services of Onion Omega2 through Nmap. During port scanning, we have identified ports 22, 53, 80, 443, 1883, 4200 and 9001 are open for the services such as TCP (transmission control protocol), SSH (secure shell), HTTP (hypertext transfer protocol), HTTPs (hypertext transfer protocol secure), Mosquito version 1.4.15, and Nagios Service Check Acceptor (NSCA), respectively. Figure 4 shows the open ports and services of Onion-Omega2 that we have identified.



OS	Host
?	192.168.3.1

	Port	Protocol	State	Name	Version
●	22	tcp	open	ssh	Dropbear sshd (protocol 2.0)
●	53	tcp	open	domain	dnsmasq 2.75
●	80	tcp	open	http	
●	443	tcp	open	https	
●	1883	tcp	open	mosquitto version 1.4.15	
●	4200	tcp	open	http	ShellInABox httpd
●	9001	tcp	open	nagios-nsca	Nagios NSCA

Figure 4. Open ports and services of Onion-Omega2.

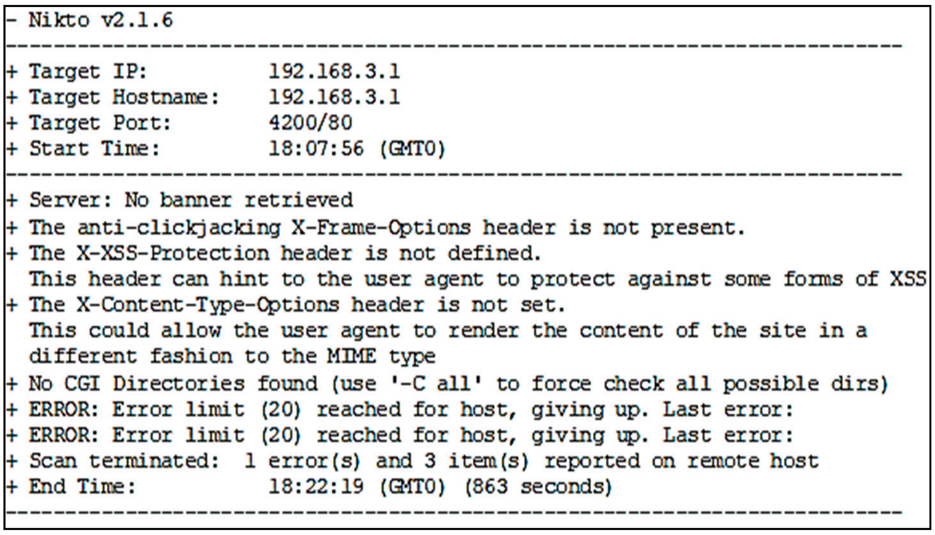
4.1.2. Web Server Assessment

By digging deeper, our next target was to test the webserver of Onion Omega2. Webserver scanning conducts a black box test to identify security vulnerabilities. Nikto is a popular Linux-based open-source web server scanner. It performs complete testing for a variety of vulnerabilities, such as version specific problems, outdated servers, missing headers as anti-clickjacking, X-Frame-Options header, dangerous files and common gateway interface (CGI) scripts. While scanning, we found that the webserver of Onion Omega2 could be at risk of a clickjacking attack owing to bad coding practices.

Clickjacking attacks are a form of cross-site request forgery that may be used for stealing important information and obtaining critical information about the application [32]. In this intrusion, the attacker uses specialized HTML content to hide elements of a webpage behind other page elements.

We found three flaws in the webserver implementation, namely, absence of X-Frame-Options header, lack of definition, X-XSS protection header and X-Content-Type-Options header not set for the targeted port 4200 and 80 (version ShelllnABox). The X-Frame-Options in the HTTP response header are used to indicate whether a browser should be allowed to render a page inside a frame or iframe. The proper usage of such options will avoid clickjacking attacks, by ensuring that their content is not embedded into other malware sites [32]. To mitigate this problem, the vendor should configure the webserver properly by including an X-Frame-Options/X-XSS protection/C-Content-Type-Options into the header [33]. In addition, it is difficult to detect such attacks, but can be avoided by using the Noscript extension for browsers like the ClearClick extension for Firefox which analyzes webpages for visually hidden elements and warns the user of suspicious content.

Moreover, while evaluating the webserver of Onion Omega2 using the OpenVAS scanner, we came across a high severity vulnerability which indicates buffer-overflow due to long URLs applied to the remote webserver on http port 4200. Buffer overflow vulnerabilities occur when an operating system or application server does not perform proper memory management. In such cases, an attacker can provide a program with carefully crafted input, and they may be able to gain remote access and/or take control of the target system. The solution to this problem, as with most other buffer overflow vulnerabilities, is to apply a proper patch as the impact of this vulnerability allows the attacker to execute the arbitrary code, which can be particularly dangerous. Figure 5 depicts the Nikto logs on Sparta indicating the webserver vulnerabilities (for clickjacking attacks) of Onion Omega2. To fix this issue, the vendor should upgrade the web server to the latest version and apply proper fixes for the buffer-overflow vulnerability.

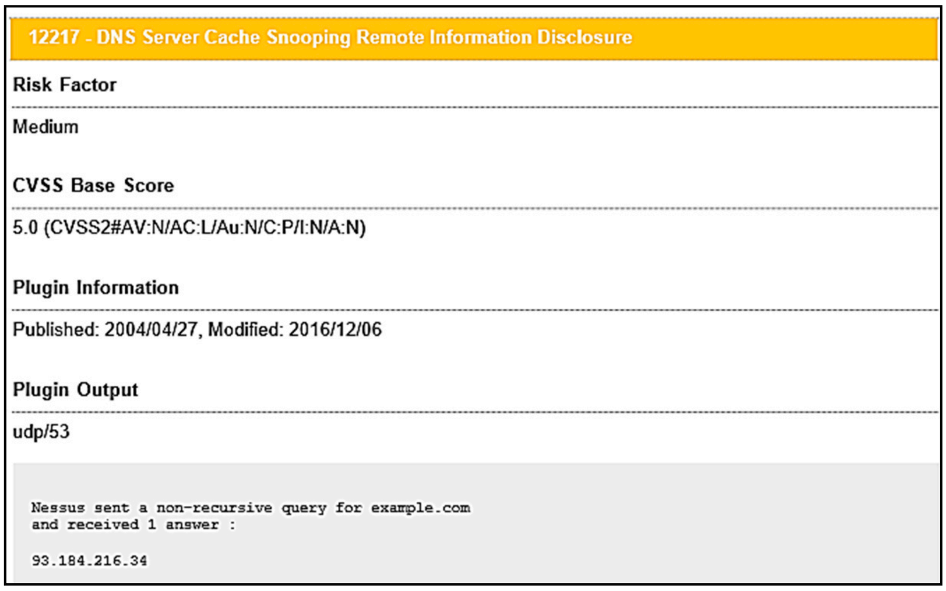


```
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.3.1
+ Target Hostname:    192.168.3.1
+ Target Port:        4200/80
+ Start Time:         18:07:56 (GMT0)
---------------------------------------------------------------------------
+ Server: No banner retrieved
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined.
  This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set.
  This could allow the user agent to render the content of the site in a
  different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ ERROR: Error limit (20) reached for host, giving up. Last error:
+ ERROR: Error limit (20) reached for host, giving up. Last error:
+ Scan terminated:  1 error(s) and 3 item(s) reported on remote host
+ End Time:           18:22:19 (GMT0) (863 seconds)
---------------------------------------------------------------------------
```

Figure 5. Webserver vulnerabilities (Nikto logs).

4.1.3. DNS Server Assessment

The remote DNS server of Onion Omega2 responded to queries for third-party domains that do not have the recursion bit set on port 53 (UDP), a dnsmasq 2.75 service. For proper validation, Nessus sent a non-recursive query to the Onion Omega2 DNS server and received the IP address 93.184.216.34. This means that the remote server is vulnerable to cache snooping attacks. Figure 6 highlights the vulnerability called "DNS Server Catch Snooping Remote Information Disclosure" with medium severity and a CVSS score of 5.0.



12217 - DNS Server Cache Snooping Remote Information Disclosure

Risk Factor

Medium

CVSS Base Score

5.0 (CVSS2#AV:N/AC:L/Au:N/C:P/I:N/A:N)

Plugin Information

Published: 2004/04/27, Modified: 2016/12/06

Plugin Output

udp/53

```
Nessus sent a non-recursive query for example.com
and received 1 answer :

93.184.216.34
```

Figure 6. DNS server cache snooping vulnerability.

As per Nessus, the severity of DNS server cache snooping remote information disclosure weakness is medium as this vulnerability may allow a remote attacker to determine which domains have recently been resolved via this name server, and therefore which hosts have been recently visited [34]. If this is an internal DNS server not accessible to outside networks, attacks will be limited to the internal network. This may include employees, consultants and potentially users on a guest network or a Wi-Fi connection [35]. To mitigate this vulnerability, the vendor should fix the problem of DNS software with proper patching. The rest of the vulnerabilities presented in Nessus report will be discussed in the subsequent section. Table 1 summarizes vulnerabilities and subsequent mitigations of Onion Omega2 fixed by the vendor.

Table 1. Onion Omega2 vulnerability and mitigation (vendor fixes).

Vulnerabilities	Description	Targeted Ports/Services	Scanning Tools Used	Mitigation
X-XSS-Protection header is not defined.	Cross-site scripting attack on index page is possible. This could allow the user agent to render the content of the site in a different fashion to the MIME type.	4200, 80 Http	Nmap, Nikto, Sparta	1. Vendor should configure the webserver including an X-Frame-Options/ X-XSS protection/X-Content-Type-Options header. 2. Fix the header protection by enabling MOD Header in httpd.conf file. 3. Enable XSS filter to sanitize the page if attack detected. 4. Use No Script extension for browser (e.g., Clear Click for Firefox)
Buffer Overflow attack possible.	Long string from remote host generates buffer overflow in memory.	4200, Http	Nmap, OpenVAS	Vendor should upgrade the web server to the latest version by adopting proper validation techniques for each segment of the program code.

Table 1. *Cont.*

Vulnerabilities	Description	Targeted Ports/Services	Scanning Tools Used	Mitigation
DNS Server Cache Snooping Remote Information Disclosure	This may allow a remote attacker to determine domains that have recently been resolved via this name server.	53-DNS (UDP/TCP) dnsmasq 2.75	Nmap, Nessus	Vendor should update the DNS server with proper patches to fix the problem.
SSH Weak MAC Algorithms Enabled	SSH server is configured to allow either MD5 or 96-bit MAC algorithms, both of which are considered weak.	22-SSH	Nessus, OpenVAS	Vendor should patch it, or administrator should consult product documentation to disable MD5 and 96-bit MAC algorithms.
Key-Reinstallation Attack (KRACK) Vulnerability [36]	This breaks security of WPA2 protocol and able to access the critical information communicating between client and Access point.	WPA2	Krackattacks-scripts [37]	Vendor should fix the issue by properly configured wpa-supplicant in Linux kernel.

Table 1 summarizes vulnerabilities and subsequent mitigations of Onion Omega2 fixed by the vendor.

According to the latest details, Onion Omega2—v0.3.2 b217 and up, have patched vulnerabilities 4 and 5 as mentioned in Table 1. However, the first three vulnerabilities listed in Table 1 are not patched yet and have been left to Onion Omega2 end users to implement the security hardening. One reason these vulnerabilities have not been patched is that the Linux build system is open-source and the firmware of Onion Omega2 is meant to be a "jack of all trades", and compatible to the end users to build their own customized firmware that can be security hardened according to their needs. The source to build the customized firmware of Onion Omega2 is available [37].

4.2. Network/Administrator Fixes

This section presents the major weaknesses of Onion Omega2 for configuring SCADA networks. We have found three major vulnerabilities of Onion Omega2 that need to be fixed at the network-administrator level. The first vulnerability relates to the IP forwarding service that is enabled in Onion Omega2, which could act as an unreliable path to bypass the firewall. The second weakness was found in configuration of the message queue telemetry transport (MQTT) broker. An MQTT broker is a lightweight protocol and is used to establish the communication with low-battery IoT devices. This protocol does not protect with an authentication password, consequently, allowing intruders to extract information of internal network. The third vulnerability includes a medium strength cipher configured for https protocol suites of the secure socket layer. In the following we discuss each of these vulnerabilities in detail. The comprehensive summary of network level fixes is provided in Table 4 (which appears later in this section).

4.2.1. IP Forwarding Service

Unless the host is a router, IP forwarding is used to forward packets from one host to another. This service must be disabled since attackers can exploit it to route the packets through the IP enabled host and bypass targeted firewalls, routers or network address control (NAC) filters [38]. In Onion Omega2, the remote host has IP forwarding enabled. Unless Onion Omega2 acts as a gateway or router in the system, it is recommended to disable IP forwarding service by setting the "IP Forward" variable to 0. However, in our proposed scheme, Onion Omega2 acts as a gateway between field devices and mobile clients and requires IP forwarding enabled. For Linux, Windows and Mac operating systems, the settings to disable IP forwarding are mentioned in Table 4. For other systems, the network administrators should contact vendors for appropriate settings.

4.2.2. SSL Medium Strength Suites

Digital certificates allow for secure exchange of public encryption keys over the control networks. Transport layer security (TLS) or secure socket layer (SSL) use these certificates to facilitate secure communication over the networks. SSL was the predecessor to TLS and works in a very similar way. However, there are known security flaws in SSL such as deflate compression, robot attack and OpenSSL heartbleed attack. Therefore, our first approach is to validate the https configuration protocol for Onion Omega2. We have configured https on Onion Omega2 and evaluated the strength of the protocol using the vulnerability assessment tool, Nikto. As mentioned in the Table 2, https is configured with the strong communication protocol suite TLS v1.2 which establishes a secure communication link between the client and the web server, which will prevent the network communication from the three major SSL attacks mentioned above.

Table 2. HTTPs (hypertext transfer protocol secure) configuration and assessment.

HTTPS Secure Communication	Assessment Outcomes
Configuration Protocol	OK—TLS V1.2 (Strong Communication protocol)
Forward Secrecy	OK—Supported
Client-initiated renegotiations	OK—Supported
OpenSSL CCS Injection	OK—Not vulnerable to OpenSSL CCS injection
Deflate Compression	OK—Compression disabled
ROBOT Attack	OK—Not vulnerable
OpenSSL Heartbleed Attack	OK—Not vulnerable to Heartbleed
Downgrade Attacks (TLS_FALLBACK_SCSV)	OK—Supported
Default Configuration:	Vulnerable to Sweet32 attacks
ECDHE-RSA-DES-CBC3-SHA2	Preferred Configuration:
Medium Strength Ciphers:	ECDHE-RSA-AES128-CBC3-SHA2
Uses 64 bits block cipher 3DES	ECDHE-RSA-AES128-GCM-SHA3

To choose the cipher suites for secure communication, the client sends a request to the web server. This request includes a list of the cipher suites supported by the client. TLS chooses the list of cipher suites which includes the encryption algorithms, hash functions, and other cryptographic details. The strength of the TLS is based on choice of algorithms included in the firmware. Hence, it is important to choose strong cipher suites from the available pool. By default, Onion Omega2 is configured with the elliptic curve Diffie Hellman algorithm for key exchange, the RSA algorithm for authentication, the 3DES-CBC algorithm for message integrity, and SHA2 for message authentication code. However, this configuration provides medium strength security due to 3DES. This symmetric key cryptography is vulnerable to SWEET32 attack [39]. Hence, while configuring, the network administrator needs to explicitly change the default configuration with the preferred configuration as listed in Table 3.

Table 3. Default configuration of transport layer security (TLS) in Onion Omega2.

Default Configuration	Preferred Configuration
ECDHE-RSA-DES-CBC3-SHA2	TLS_RSA_WITH_AES_128_GCM_SHA384
ECDHE: Elliptic Curve Diffie-Hellman Key Exchange	(GCM uses parallel approach)
RSA: for Authentication	TLS_RSA_WITH_AES_256_CBC_SHA256
3DES-CBC: Symmetric Encryption	(CBC 256 bits consist of 14 rounds)
SHA2: Message Authentication Code	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
vulnerable to SWEET32 attack (CVE-2016-2183)	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA256

4.2.3. MQTT Broker Protocol

Message queue telemetry transport (MQTT) protocol functions on top of the transport layer protocols. Generally, it is used in communication amongst IoT based devices. This lightweight protocol was created by IBM and standardized by ISO/IEC 20922 specifically for low-bandwidth environments [40]. The MQTT protocol supports authentication, which prevents the system from unauthorized access control. The MQTT protocol should be configured with a proper authentication password and must also run on the SSL/TLS layer to protect the communication data from attackers [41]. The default configuration of the MQTT protocol in Onion Omega2 is not protected with a password; that can expose the entire control system to the outside world. Therefore, in general, it is required to set up the proper password by enabling the authentication mechanism before configuring the MQTT broker in the network. However, in this use case MQTT is not involved and hence we have disabled the Mosquito service during configuration.

Table 4 summarizes vulnerabilities and subsequent mitigations of Onion Omega2 fixed by the network administrators.

Table 4. Onion Omega2—network administrator fixes. message queue telemetry transport = (MQTT).

Vulnerabilities	Description	Scanning Tools	Mitigation
IP forwarding enabled	An attacker can exploit this path to route packets through the specific host to bypass targeted firewalls, routers or NAC filters.	OpenVAS, Nessus	1. Unless the remote host is a router/gateway, it is recommended to disable IP forwarding. 2. On Linux based firmware, disable the IP forwarding by doing [38]: echo 0 > /proc/sys/net/ipv4/ip_forward 3. On Windows based system, set the key 'IPEnableRouter' to 0 under [38]: HKEY_LOCAL_MACHINE\System \CurrentControlSet\Services\Tcpip \Parameters 4. On Mac OS X, to disable IP forwarding execute following command [38]: sysctl -w net.inet.ip.forwarding = 0
SSL medium strength cipher suites supported	The default configuration supports encryption that uses 3DES-CBC for symmetric key cryptography which is vulnerable to SWEET32.	Nessus, OpenVAS, Nikto	Reconfigure the affected application to avoid use of medium strength ciphers [39].
MQTT Broker does not required authentication	This may allow a remote attacker to gain access of the network without any authentication. Intruder can extract the information of the control/IoT networks.	OpenVAS	Enable authentication mode in Onion Omega2: By configuring Mosquitto Broker setting password in etc/mosquito/Passwd;

5. Generic Vulnerability Assessment

This paper mainly focused on the vulnerability assessment of Onion Omega2. However, the procedure that we have adopted can be useful for the assessment and analysis of other platforms. Vulnerability assessment is a continuous process due to constant technological changes and hence becomes the backbone for a successful defense of any industrial control system. This process is heavily dependent on asset management and risk assignment to prioritize security issues. Figure 7 depicts a generic flowchart for vulnerability analysis, in which we generalize the assessment process for other devices, platforms, and networks in industrial control systems. The proposed strategy can be used to maintain the security and compliance standards of the system with open-source tools and technologies.

Using this generic framework, vendors, security analysts and network engineers can build their own configuration and run different scans to detect security flaws of the system. Our approach gives the ability to assess the infrastructure thoroughly, which covers different levels such as host, network, wireless, and application-level vulnerabilities.

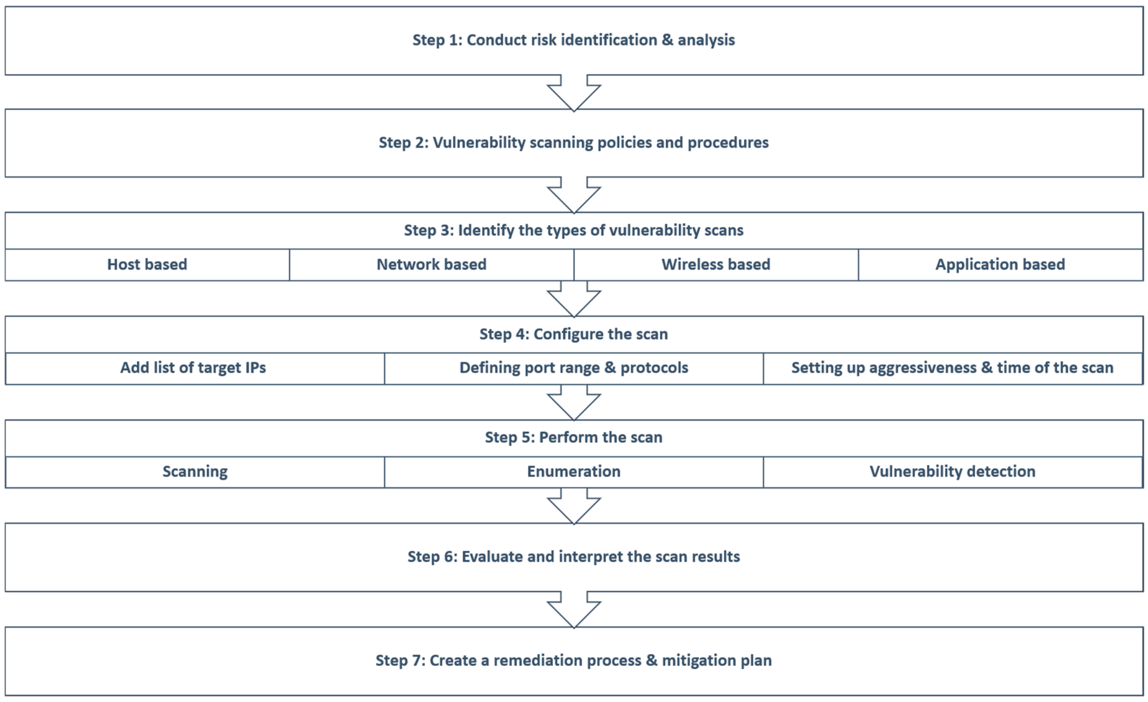

Step 1: Conduct risk identification & analysis
Step 2: Vulnerability scanning policies and procedures
Step 3: Identify the types of vulnerability scans
Host based
Network based
Wireless based
Application based
Step 4: Configure the scan
Add list of target IPs
Defining port range & protocols
Setting up aggressiveness & time of the scan
Step 5: Perform the scan
Scanning
Enumeration
Vulnerability detection
Step 6: Evaluate and interpret the scan results
Step 7: Create a remediation process & mitigation plan

Figure 7. Generic framework of vulnerability assessment process.

6. Conclusions

The Onion Omega2 is a small embedded Linux server for building SCADA/IoT communication systems. While it provides efficient functionality, it is important to be aware of its vulnerabilities and built-in security features. We have identified product level vulnerabilities of Onion Omega2 using scanners and penetration tools. This helped us to identify the threats and vulnerabilities of Onion Omega2 and measure the level of risk. The vulnerabilities include missing patches, insecure system configurations, and other security-related updates. The identified vulnerabilities can either be fixed by the vendor and/or network administrator/engineer. Furthermore, the paper illustrates effective countermeasures for identified vulnerabilities to harden the security of Onion Omega2. This study empowers vendors, software developers and network engineers with the knowledge necessary to take proactive measures to ensure security of the overall system built using Onion Omega2. The scope of further study includes similar analysis for other popular microcontrollers such as Raspberry PI, Beagle bone and Arduino.

Author Contributions: D.U.: conceptualization, formal analysis, methodology, writing—first draft; S.S.: funding acquisition, supervision, writing—review and editing; B.P.: project administration. All authors have read and agreed to the published version of the manuscript.

Funding: The authors gratefully acknowledge the support in part by the Natural Sciences and Engineering Research Council (NSERC) and industry partners Cistel Technology Inc, through a Collaborative Research Grant.

Acknowledgments: Thanks to Brian Stacey, Rohit Joshi and Marzia Zaman of Cistel Technologies for their valuable feedback.

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

References

1. Choo, K.-K.R.; Gritzalis, S.; Park, J.H. Cryptographic solutions for industrial Internet-of-Things: Research challenges and opportunities. *IEEE Trans. Ind. Inf.* **2018**, *14*, 3567–3569. [CrossRef]
2. Garcia-Morchon, O.; Kumar, S.; Sethi, M. Internet of Things (IoT) Security: State of the Art and Challenges. 2019. Available online: https://www.rfc-editor.org/rfc/pdfrfc/rfc8576.txt.pdf (accessed on 8 May 2020).
3. Ramos, A.; Lazar, M.; Filho, R.H.; Rodrigues, J.J.P.C. Model-Based Quantitative Network Security Metrics: A Survey. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 2704–2734. [CrossRef]
4. Ghaleb, A.; Zhioua, S.; Almulhem, A. On PLC network security. *Int. J. Crit. Infrastruct. Prot.* **2018**, *22*, 62–69. [CrossRef]
5. Cyber-Attack on Indian Nuclear Power Plant Exposes Threat of "Snooping" Malware. Available online: https://www.bitsight.com/blog/cyber-attack-on-indian-nuclear-power-plant-exposes-threat-of-snooping-malware (accessed on 12 November 2019).
6. National Cybersecurity and Communications Integration Center/Industrial Control Systems Cyber Emergency Response Team. *NCCIC/ICS-CERT FY 2015 Annual Vulnerability Coordination Report*; Homeland Security: Washington, DC, USA, 2015.
7. Common Vulnerabilities Exposures Database CVE – Home. Available online: https://cve.mitre.org/cve/ (accessed on 2 October 2018).
8. Cherdantseva, Y.; Burnap, P.; Blyth, A.; Eden, P.; Jones, K.; Soulsby, H.; Stoddart, K. A review of cyber security risk assessment methods for SCADA systems. *Comput. Secur.* **2016**, *56*, 1–27. [CrossRef]
9. Onion – Compute Platform for IoT. Available online: https://onion.io/ (accessed on 8 June 2020).
10. Upadhyay, D.; Sampalli, S. SCADA (Supervisory Control and Data Acquisition) Systems: Vulnerability Assessment and Security Recommendations. *Comput. Secur.* **2020**, *89*, 101666. [CrossRef]
11. Nicholson, A.; Webber, S.; Dyer, S.; Patel, T.; Janicke, H. SCADA security in the light of Cyber-Warfare. *Comput. Secur.* **2012**, *31*, 418–436. [CrossRef]
12. Horkan, M. *Information Security Reading Room A Black-Box Approach to Embedded Systems*; The SANS Institute: Bethesda, MD, USA, 2016.
13. Onion Omega2 Known Firmware Issues | Onion Omega2 Documentation. Available online: https://docs.onion.io/omega2-docs/known-firmware-issues.html (accessed on 27 May 2020).
14. Cagalaban, G.; Kim, T.; Kim, S. Towards Improving SCADA Control Systems Security with Vulnerability Analysis Improving SCADA Control Systems Security with Software Vulnerability. In Proceedings of the International Conference on Parallel and Distributed Computing and Networks, Chongqing, China, 13–14 December 2010.
15. Coffey, K.; Smith, R.; Maglaras, L.; Janicke, H. Vulnerability Analysis of Network Scanning on SCADA Systems. *Secur. Commun. Netw.* **2018**, *2018*. [CrossRef]
16. Andersson, S.; Josefsson, O. On the assessment of Denial of Service vulnerabilities Affecting Smart Home Systems: Vulnerability Scanning Using OpenVAS. 2019. Available online: https://muep.mau.se/handle/2043/29162 (accessed on 8 May 2020).
17. Andrews, M. The state of web security. *IEEE Secur. Privacy* **2006**, *4*, 14–15. [CrossRef]
18. Steven, J. Adopting an enterprise software security framework. *IEEE Secur. Privacy* **2006**, *4*, 84–87. [CrossRef]
19. Samtani, S.; Yu, S.; Zhu, H.; Patton, M.; Matherly, J.; Chen, H. Identifying SCADA systems and their vulnerabilities on the internet of things: A text-mining approach. *IEEE Intell. Syst.* **2018**, *33*, 63–73. [CrossRef]
20. Sun, Y.; Sun, L.; Shi, Z.; Yu, W.; Ying, H. Vulnerability finding and firmware association in power grid. In Proceedings of the 2019 5th International Conference on Mobile and Secure Services, MOBISECSERV 2019, Miami Beach, FL, USA, 2–3 March 2019; pp. 1–5. [CrossRef]
21. Carranza, A.; Mayorga, D.; DeCusatis, C.; Rahemi, H. Comparison of wireless network penetration testing tools on desktops and raspberry Pi platforms. In Proceedings of the LACCEI international Multi-conference for Engineering, Education and Technology, Boca Raton, FL, USA, 18–20 July 2018; pp. 19–21. [CrossRef]
22. Binwalk Binwalk | Penetration Testing Tools. Available online: https://tools.kali.org/forensics/binwalk (accessed on 3 November 2019).
23. Leszczyna, R. Approaching secure industrial control systems. *IET Inf. Secur.* **2015**, *9*, 81–89. [CrossRef]
24. Maiolo, M.; Pantusa, D. Infrastructure Vulnerability Index of drinking water systems to terrorist attacks. *Cogent Eng.* **2018**, *5*, 1456710. [CrossRef]

25. Department of Energy. *Enhancing Control Systems Security in the Energy Sector NSTB National SCADA Test Bed Common Cyber Security Vulnerabilities Observed in Control System Assessments by the INL NSTB Program*; Department of Energy: Washington, DC, USA, 2008.
26. Van Der Maelen Uría, J.F.; Alvarez-Rúa Alvarez, C. Using Spline Functions for Obtaining Accurate Partial Molar Volumes in Binary Mixtures. *Comput. Chem.* **1998**, *22*, 225–235. [CrossRef]
27. Nurse, J.R.C.; Creese, S.; De Roure, D. Security risk assessment in Internet of Things systems. *IT Prof.* **2017**, *19*, 20–26. [CrossRef]
28. Jing, Q.; Vasilakos, A.V.; Wan, J.; Lu, J.; Qiu, D. Security of the Internet of Things: Perspectives and challenges. *Wirel. Netw.* **2014**, *20*, 2481–2501. [CrossRef]
29. Yang, L.; Yang, S.-H.; Yao, F. Safety and security of remote monitoring and control of intelligent home environments. In Proceedings of the 2006 IEEE International Conference on Systems, Man and Cybernetics, Taipei, Taiwan, 8–11 October 2006; pp. 1149–1153.
30. Ali, B.; Awad, A.I. Cyber and physical security vulnerability assessment for IoT-based smart homes. *Sensors* **2018**, *18*, 817. [CrossRef] [PubMed]
31. Cohen, M.; Graf, R.; Kilani, M.; Pashley, C.; Ryan, J.; Shannon, G.; Walters, G.; Wills, T. Cyber Resilience and Response 2018 Public-Private Analytic Exchange Program. 2018. Available online: https://www.dhs.gov/sites/default/files/publications/2018_AEP_Cyber_Resilience_and_Response.pdf (accessed on 8 May 2020).
32. Acunetix Clickjacking: X-Frame-Options Header Missing - Vulnerabilities – Acunetix. Available online: https://www.acunetix.com/vulnerabilities/web/clickjacking-x-frame-options-header-missing/ (accessed on 8 December 2019).
33. OWASP. OWASP Secure Headers Project. Available online: https://www.owasp.org/index.php/OWASP_Secure_Headers_Project (accessed on 8 December 2019).
34. Tenable Nessus DNS Server Cache Snooping Remote Information Disclosure | Tenable®. Available online: https://www.tenable.com/plugins/nessus/12217 (accessed on 11 November 2019).
35. Grangeia, L. DNS Cache Snooping Snooping the Cache for Fun and Profit. Available online: http://www.rootsecure.net/content/downloads/pdf/dns_cache_snooping.pdf (accessed on 11 November 2019).
36. Vanhoef, M.; Piessens, F. Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2. In Proceedings of the ACM Procedding; Publisher:Association for Computing Machinery, Dallas, TX, USA, 17–20 October 2017.
37. Onion Omega2, S. GitHub - OnionIoT/Source: Onion's fork of OpenWRT's Source Build System. The Firmware for the Omega2, Omega2+, and Omega2 Pro is Based on the Openwrt-18.06 Branch. Available online: https://github.com/OnionIoT/source (accessed on 1 November 2019).
38. Tenable Nessus IP Forwarding Enabled | Tenable®. Available online: https://www.tenable.com/plugins/nessus/50686 (accessed on 11 November 2019).
39. Tenable Nessus. *Nessus Report*; Tenable: Columbia, MD, USA, 2009.
40. Dinculean, D.; Cheng, X. Vulnerabilities and Limitations of MQTT Protocol Used between IoT Devices. *Appl. Sci.* **2019**, *9*, 848. [CrossRef]
41. Crawley, S. MQTT Broker Does Not Require Authentication – QRIScloud. Available online: https://support.qriscloud.org.au/hc/en-us/articles/360000375956-MQTT-Broker-does-not-require-authentication (accessed on 11 November 2019).