Regular Paper

Empirical Analysis of Security and Power-Saving Features of Port Knocking Technique Applied to an IoT Device

Aamir H. Bokhari[1,a)] Yuta Inoue[1] Seiya Kato[1] Katsunari Yoshioka[1,2]
Tsutomu Matsumoto[1,2]

Received: November 30, 2020, Accepted: June 7, 2021

Abstract: The digital boom brought empowerment to seamless connectivity by enabling manufacturers to harness the power of the Internet into their products, opening up the world of the Internet of Things (IoT). However, such connectivity has also brought the side effect of such power being abused by unscrupulous agents, who scan open ports for services and exploit vulnerabilities in the system. The Mirai botnet malware attack is one such example that caused havoc by compromising millions of IoT devices having unpatched/weaker security. There is an increasing need to enable IoT devices to be fully patched and secured, but such methods are often under attack. This paper examines a stealth technology and its impact on the CPU and power consumption to secure resource-constraint IoT devices that are growing exponentially. By enabling secure remote operations and management of such devices using a unique but practical method of security called "Port Knocking," we can ensure timely patching of security vulnerabilities in a safe and stealthy manner. Our experimental results on a resource-constraint IoT device show that port knocking not only secures the device and provides a secure remote management option but also helps in keeping its power consumption low. The results obtained make it an effective security layer for securing resource-constraint IoT devices.

Keywords: authentication, IoT security, port knocking, remote management

1. Introduction

In the 21st century, the Internet of Things (IoT) has opened up a new horizon for entrepreneurs and hackers. According to a well-known security company "Norton," the number of IoT devices is estimated to reach 21 billion by 2025 [1]. The booming 5G technology means billions of IoT devices can connect directly to the Internet using the 5G speeds over the cellular networks [2], which would make them more susceptible to direct Internet attacks [1]. The current situation in the year 2020 is that Gartner expects over 25% of known attacks to involve IoT, whereas the IT security budgets for IoT would be less than 10% [3]. The IoT device resources are also getting scarce due to small sizes designed for portable use. This may result in fewer or almost no traditional security features due to the resource constraints in the IoT devices. These issues will increase the opportunities for hacking [4], [5]. Furthermore, botnets are using self-propagating malware, such as "Mirai" for attack purposes. The first large scale attack on a single enterprise was witnessed in the year 2016 that used millions of compromised IoT devices by the "Mirai" botnet malware, the code of which is now available on the Internet for anyone to use or modify [1], [6], [7]. Therefore, with the growth of the IoT market, the attack field is equally broadening and cyber-attacks exploiting IoT vulnerabilities in network services and inadequate security are on the rise. On the other hand, the end-users do not have adequate technical knowledge to fix security issues by themselves. The lack of security management and limited resources on IoT devices has, therefore, become a big challenge.

One possible approach is to keep the security of IoT devices up-to-date using the remote management feature. But, the research shows that such features are also subject to common attacks on well-known services, such as Telnet, FTP, SSH, and Web [1], [5], [6], [8], [9]. These remote management services are among the top 20 most scanned ports and often are an ideal target for exploitation as they provide direct access to the system with escalated privileges [10]. In order to secure the remote management capability, it would require incorporating additional security measures on the already resource-constrained IoT devices. Conventional security methods, such as firewall white-listing or VPN are resource consuming, and therefore, are often used at the enterprise level or among computing devices that have a lot more resources available than the resource-constrained IoT devices. Also, such perimeter defenses usually require technical knowledge to configure and maintain those high-end computing devices with appropriate security rules and patches. In the case of IoT devices, such conventional security methods are not so practical to use due to the limitation of available resources and the lack of user knowledge. Thus, a lightweight, secure solution for remote management is required in IoT devices. Port knocking is one such technique in which a port can be configured to remain hidden (closed) until receiving a predetermined set of knocks (packets) on different ports in a specific order. Re-

[1] Graduate School of Environment and Information Sciences, Yokohama National University, Yokohama, Kanagawa 240–8501, Japan
[2] Institute of Advanced Sciences, Yokohama National University, Yokohama, Kanagawa 240–8501, Japan
[a)] aamir-bokhari-rd@ynu.jp

search literature shows that studies have been conducted on various port knocking methods often used for the remote management of large systems and importing them into the IoT world as well [11], [12], [13], [14], [15]. However, such studies have been mainly focused on port knocking algorithms, authentication, and various attacks or complexity of an algorithm being used for knocking and its impact on the performance in terms of protocols, physical memory, or network bandwidth. Also, security testing has often been only in a controlled environment using penetration testing and not exposing the IoT device with port knocking feature directly to the Internet over the cellular network. Existing research is missing an important piece of information that can greatly impact the intended use of an already resource-constraint IoT device. We must also examine the impact of a security feature on CPU usage and power consumption to ensure that the additional security features do not impact negatively on the resource-constraint IoT devices. These elements are vital as IoT devices footprint is becoming smaller and smaller along with the diversity in usage. Therefore, to the best of our knowledge, the below research questions remain unclear:

(1) How much computing power (CPU consumption) would the port knocking security feature add to an already resource-constraint IoT device?

(2) With respect to the high-speed cellular connectivity of IoT devices, how effective would this security feature be in blocking unwanted access to the protected service when the IoT device is exposed to the Internet directly without other security layers or a firewall?

(3) What would be the impact of adding the stealth port knocking security feature on the power consumption of the IoT device?

In order to find answers to the above research questions, we narrowed our scope to focus on IoT devices with cellular connectivity, such as those used in smart cities, smart bicycles, goods tracking, flood monitoring, agriculture monitoring, medical monitoring, wearables, etc. Such use cases need cellular IoT in order to ensure mobility and coverage for their intended use. We initially experimented for a short period (2 weeks) to obtain preliminary results and then later on tested over a longer period (7 weeks) to verify the results. An IoT device was used to test two types of port knocking methods. The first method was based on the python script applying the pseudo-random number generator (PRNG) and the chaotic random number generator (CRNG) algorithms [8]. The second method was also based on the python script, but applying a stream cipher using the Authenticated Encryption with Associated Data (AEAD) algorithms [15]. The details of testing and analysis are provided in Section 3. Based on our test results, we can summarize the answers to our research questions as follows:

(i) The CPU consumption test showed that when the stealth port knocking feature was implemented using the stream cipher with AEAD algorithm on the IoT device, it would add a maximum overhead of 15% of the CPU power. Whereas, the PRNG-CRNG algorithm would add an overhead of 50% of the CPU power.

(ii) The security effectiveness test (by exposing directly to the Internet via 3G without any other security layers) showed that the IoT device with the stealth feature protecting the SSH service running on the default port was able to block all unwanted accesses for 42 days, while the unprotected services received 431,142 requests from 5,424 hosts in these 42 days.

(iii) The power consumption test showed that the IoT device with the stealth port knocking feature would actually decrease power consumption compared to the one without such a feature because of receiving a lesser number of packets due to the hidden service. The gap between them slowly increases with time as we observed an average of 0.12 W difference in the first week, but then it grew to an average of 0.25 W in later weeks due to longer exposure of visible SSH default port to the Internet.

Hence, based on the results of the experiments conducted to measure the effect of using the stealth security feature of port knocking on the resource-constraint IoT device, we can conclude that the experimental results imply that the power consumption overhead by receiving incoming session requests (from scanners/malware on the Internet) without port knocking would easily exceed the power consumption for running the port knocking service. Thus, running the stealth port knocking service would be beneficial in terms of not only security enhancement but also of the power consumption on a resource-constraint IoT device.

2. Related Work

Remote management methods have always been an ideal target for hacking into a system. Research papers [16], [17], [18], [19], [20] talk about a stealthy method of port knocking that is commonly used by the system administrators of large systems for avoiding attacks on remote management services. With the increase in attacks on IoT devices, the focus has turned towards the port knocking feature and finding the kind of algorithms that can be used in the Internet of Things environment. Due to the popularity of IoT, a lot of research has been directed towards IoT threats, vulnerabilities, attacks, limitations, authentication, and challenges [4], [5], [6], [14], [15], [21]. Many researchers have also examined the security of IoT and possible countermeasures. For example, paper [4] highlights the issues with IoT devices and various types of IoT attacks along with possible countermeasures. It rules out the use of conventional cryptography in small IoT devices or limits it due to resource constraints. In paper [21], the authors introduce port knocking based on digital certificates for strengthening the authentication between the IoT devices. Another paper [9] proposed an architecture of SSH honeypot based on port knocking and intrusion detection systems for protecting a server. The concept looks similar to our approach, but this study was not focused on resource constraint IoT devices. The focus was rather on honeypots and attacks on SSH service. However, this paper strengthens the idea that port knocking can be used for security purposes. The authors in paper [14] provided a port knocking approach utilizing symmetric key encryption, Message Authentication Code (MAC), and an encrypted keep-alive system is used to secure the service port. However, it is limited to TCP ports with static IP configuration only and was tested using a

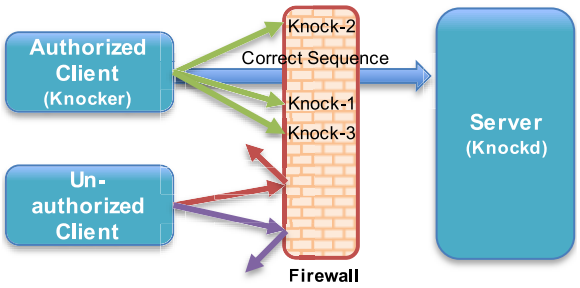


Fig. 1 Port knocking mechanism.

hardware (CPU = Intel Core i7 3770 @ 3.40 GHz; RAM = 8 GB DDR3) having plenty of resources. With respect to resources, only physical memory usage and network bandwidth were examined. This paper did not look at device power consumption, which is more critical in today's resource-constraint IoT devices.

According to another paper on security intelligence for the Industry 4.0 revolution [22], the demand for more direct connectivity with the Internet will also open doors for major security management issues. The authors have discussed possible mitigation techniques for raspberry-pi-based IoT devices using port knocking with two-factor authentication as part of their solution. However, in this study port knocking was implemented on a router and the IoT devices were behind the firewall. In paper [8], the author has introduced an advanced method of port knocking that can reduce attacks by producing difficult-to-guess port knocking sequences based on PRNG and CRNG algorithms. As this solution utilizes both TCP and UDP ports, therefore, it can be a candidate for our purposes. Similarly, another paper [15] suggests a different algorithm based on RFC7539 [23] that pairs the ChaCha20 stream cipher with the Poly1305 authenticator to create an AEAD scheme for use in the TLS protocol for high-speed and lightweight IoT applications. A security analysis report by KDDI Research Inc. concludes that they could not find a weakness in the AEAD algorithm [24]. Therefore, this method can also be a candidate for our purpose as it can be used for creating random port knocking sequences and key generation.

2.1 Port Knocking

Port knocking is not a new concept as it has been used by the system administrators to manage the servers remotely. However, its application on IoT devices is relatively new. A common method of attack is to first look for open service ports using a port scanner. In the case of port knocking, the service port is closed by default and opens for service only for the requester that sends the right sequence of packets to the firewall for authentication [25], as shown in **Fig. 1** that was created by referring to the paper [25].

In the case of Linux-based devices, a "knockd" daemon process (server) monitors the knock sequence and open/close ports via the "iptables". This knock sequence must be randomly generated and synchronized between the client and the server in order to avoid replay attacks or man-in-the-middle (MITM) type of attacks. With the service port closed, the target port cannot be confirmed during scanning, and therefore, an attacker cannot target it [25]. This stealthy feature of port knocking also helps in putting up deterrence against zero-day attacks [22].

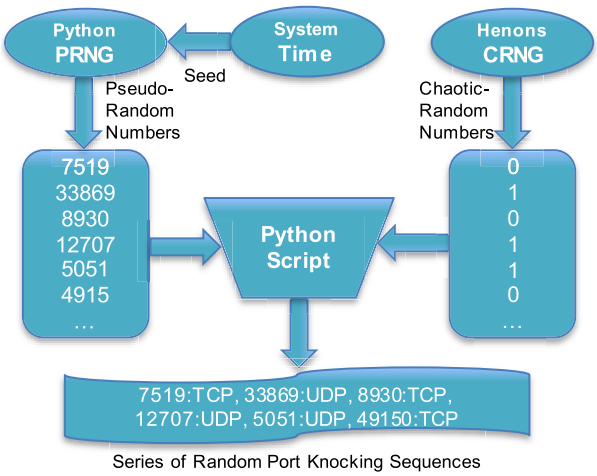


Fig. 2 PRNG-CRNG based port knocking.

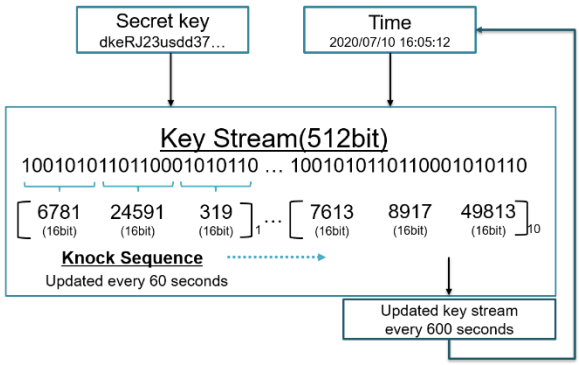


Fig. 3 Stream-cipher-based port knocking.

Based on the research work in papers [8] and [15], we selected two approaches for our port knocking test on the IoT device because they are lightweight, can be used in the IoT environment, and can generate completely random numbers using two sets of completely different algorithms as follows:

2.2 PRNG-CRNG based Port Knocking Daemon

In this approach, the port knocking mechanism (python script-based "knockd" server) is set up by a Python script that produces pseudo-random port numbers using the system time as a seed with a PRNG algorithm, and then again combining the result with the CRNG algorithm for producing random order of assigning protocol (TCP = 0 and UDP = 1) selections, as shown in **Fig. 2** that was created by referring to the paper [8].

This creates a system that always generates the same knock sequence based on the same algorithms running on the server and the client side. Due to the pseudo-random nature of the algorithms and the stealthy way of hiding the service ports, this method of port knocking mitigates denial of service (DoS), playback, and MITM attacks as well.

2.3 Stream-Cipher-based Port Knocking Daemon

The second method is also python-based "knockd" server, but it uses ChaCha20 stream cipher with Poly1305 authenticator [15]. A 256-bit secret key is shared between server and client, generating a 512-bit key stream from the secret key and time using the "Authenticated Encryption with Associated Data" algorithm, as

defined in IETF's RFC 8439 [26]. The key-stream is split into 10 knock sequences (1 sequence = 3 ports = 48 bits). The frequency for updating the knock sequence is set at 60-sec intervals. The key-stream is regenerated every 600-sec (10 min) intervals [27], as shown in **Fig. 3**.

3. Proposed Method

In order to test the stealth port knocking feature for our research, we selected a commonly available off-the-shelf IoT device from the Japanese manufacturer "Plat'Home" [28], [29], having a raspberry-pi hardware configuration with a 3G option for a direct Internet exposure using cellular network connectivity, as shown in **Table 1**. Raspberry-pi is one of the popular off-the-shelf hardware devices, supporting numerous IoT uses and applications, including as an IoT gateway for connecting various kinds of sensors with applications.

3.1 Experiment Setup

The test equipment used and the way the experiment was conducted in the lab can be seen in the photo, as shown in **Fig. 4**. Various combinations of test setups were used to make sure we can minimize any external influences on the power consumption measurements and can obtain maximum possible accuracy.

3.1.1 Test-1: Port Knocking Effectiveness in Terms of CPU Usage

First, we need to select a port knocking method that does not put too much stress on the CPU of the IoT device. We installed and tested both approaches (PRNG-CRNG based port knocking and stream-cipher-based port knocking) on the same IoT test device one-by-one, targeting port 22/TCP for SSH service with port knocking. We used the "dstat" command while connected to the Internet via a 3G line for measuring the CPU usage in each case using a log analyzer, as shown in **Fig. 5**.

3.1.2 Test-2: Port Knocking Effectiveness in Terms of Security

Once we have identified an efficient port knocking method, then we examined how effectively it could reduce the unauthorized SSH login attempts on a default port 22/TCP by setting up the test as shown in **Fig. 6**. We used two IoT devices of the same model and specifications. We installed the stealth port knocking feature on one of them to hide the SSH service running on the default port. We kept the other device running without the port knocking feature so that we can compare the difference when both are exposed directly to the Internet with the same 3G network provider.

3.1.3 Test-3: Port Knocking Effectiveness in Terms of Power Consumption

In order to examine the power consumed with and without the port knocking feature, we used the same two identical IoTBX1 devices that were running on the same software and hardware. USB testers were connected to each IoT device, and these testers were then connected to a self-powered USB hub. A note PC was also connected with the USB hub for observational purposes, as shown in **Fig. 7**. We then calibrated the USB tester without con-

Table 1 IoT test devices.

		VX2	BX1
CPU	Model	Intel Atom E3805	Intel Atom® Processor
	Speed	1.33 GHz	500 MHz
	Cache	1024 KB	1024 KB
Memory		2 GB	1 GB
Storage		32 GB	4 GB

Fig. 4 Lab experiment.

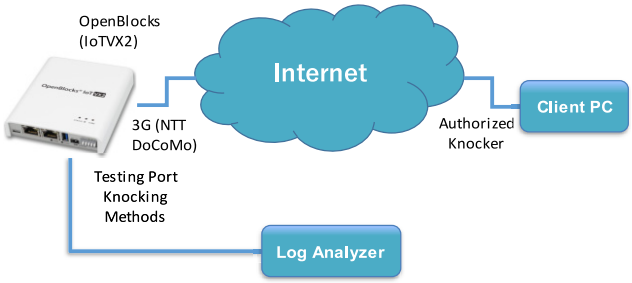


Fig. 5 Test-1 setup.

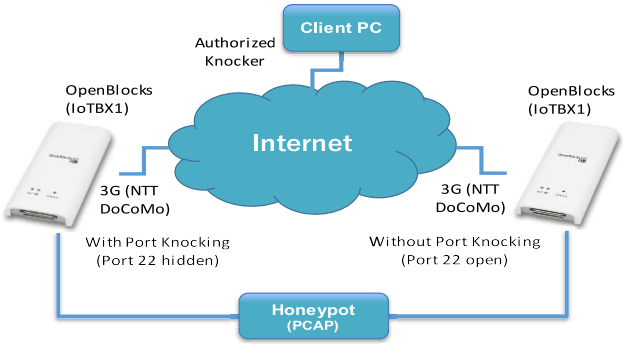


Fig. 6 Test-2 setup.

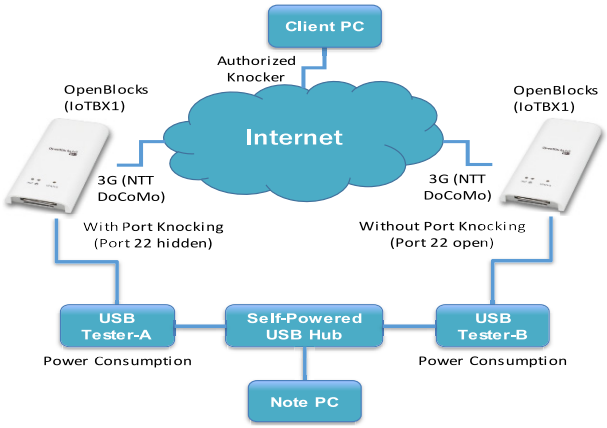


Fig. 7 Test-3 setup.

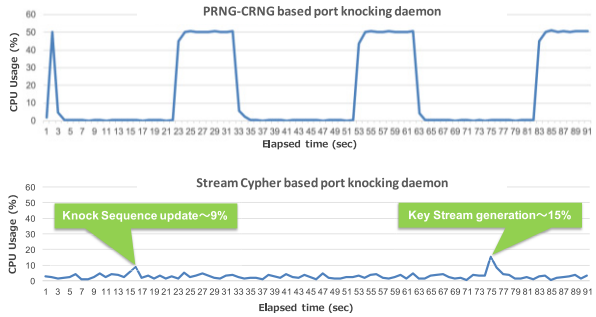


Fig. 8 Test-1 results.

necting the two IoT devices for 24 hrs. and calibrated the two IoT devices without running the port knocking feature for 24 hrs. After the benchmarking, the stealth port knocking feature was enabled on the IoT device connected to the USB Tester-A. Whereas the USB Tester-B was used to record the power consumption of the IoT device without the port knocking feature. Both were directly connected to the Internet over the 3G cellular connection from the same service provider. This way, we ensured that we could take measurements at the same time on both devices for comparison purposes.

3.2 Experiment Results

In the case of test-1 for finding out the effectiveness of port knocking in terms of CPU usage, we found that the PRNG-CRNG based port knocking method mostly consumed 50% of the CPU resources while updating the knock sequence and generating the key. Whereas in comparison, the stream-cipher-based port knocking method only consumed 9% of the CPU when updating the knock sequence, and the CPU usage was only 15% when generating the key, as shown in **Fig. 8**.

Therefore, further testing was done based on the second method (stream cipher) to confirm the effectiveness of the stealth port knocking feature by hiding the SSH service running on the default port 22/TCP (when not in use but exposed to the Internet) and the power consumption. For further testing, we used the IoTBX1 model as it has more resource constraints than IoTVX2.

In the case of test-2, the stream-cipher-based port knocking method was further tested for 6 weeks (42 days) from October 18, 2020 to November 29, 2020. The results showed that this method of port knocking is quite effective from the security point of view as it stealthily used the SSH service running on the port 22/TCP without any issue while the IoT device was directly exposed to the Internet through the 3G cellular connection without any additional security elements for 6 weeks. During this time, we observed not a single unauthorized SSH login attempt on the IoT test device running the stealth port knocking feature. This is due to the fact that port 22/TCP was hidden and did not respond to any unauthorized SYN packets it received. In comparison, the device with no port knocking feature had 431,142 SSH login attempts from 5,424 unique IP sources (hosts) due to its visible SSH service running on the default port, as shown in **Table 2**. This demonstrates that the stream-cipher-based port knocking was able to reduce the attack surface significantly, adding to the security of the IoT device.

Table 2 Port knocking security effectiveness (6 weeks test results).

	With Port Knocking	Without Port Knocking
SSH Login Attempts	0 Times	431,142 Times
Source IP Addresses	0 Hosts	5,424 Hosts

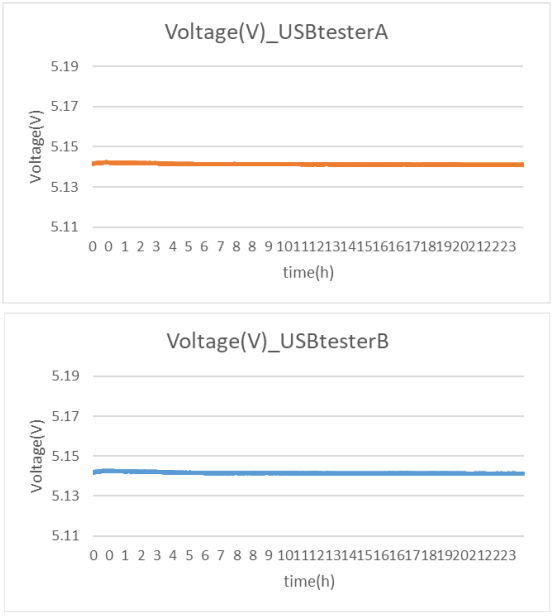


Fig. 9 Voltage stability results.

Next, we look at the test-3 results for assessing power consumption with and without the port knocking feature.

(a) Voltage Stability: In order to benchmark and make sure we do not observe any other influence on the voltage being measured by the USB testers, we connected only the two USB testers to the USB hub and measured voltage for 24 hours on July 22–23, 2020. Both came out to be stable around 5.14 volts, as shown in **Fig. 9**.

(b) Confirming power consumption without port knocking: Next, we established the baseline (benchmark) by measuring the power consumption of both IoT devices without port knocking for 24 hours on July 24–25, 2020. This way, we can observe how these IoT devices are consuming power with and without the stealth port knocking feature running on any of them. We observed almost similar readings (1.38 W on Tester-A connected IoTBX1 and 1.37 W on Tester-B connected IoTBX1), as shown in **Fig. 10**.

(c) Confirming power consumption with port knocking: After ensuring we have stable readings without the port knocking, we then enabled stream-cipher-based port knocking on the IoT device connected to the USB Tester-A only. We measured the power consumption of both IoT devices for over seven weeks (52 days), from August 3 thru 10, 2020 and from October 16 thru November 30, 2020. We also observed the total number of packets received by each IoT device by running the "netstat–statistics" command every hour. The results showed that the power consumption is directly proportional to the packets received. Since the IoT device connected to the USB Tester-A was running the port knocking feature, therefore, the number of packets received

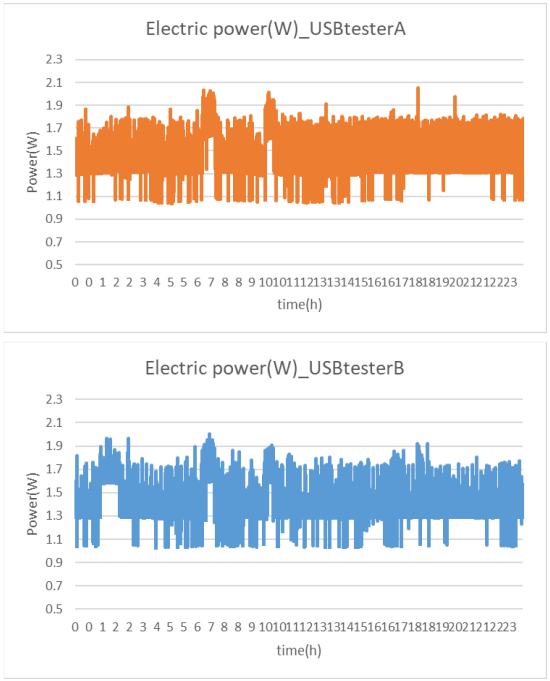


Fig. 10 Power consumption without port knocking on both test devices.

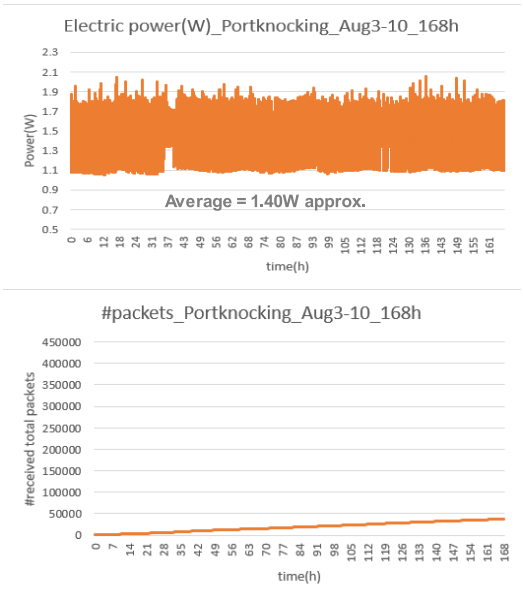


Fig. 11 Power consumption and packets received **with** port knocking during the first week of testing.

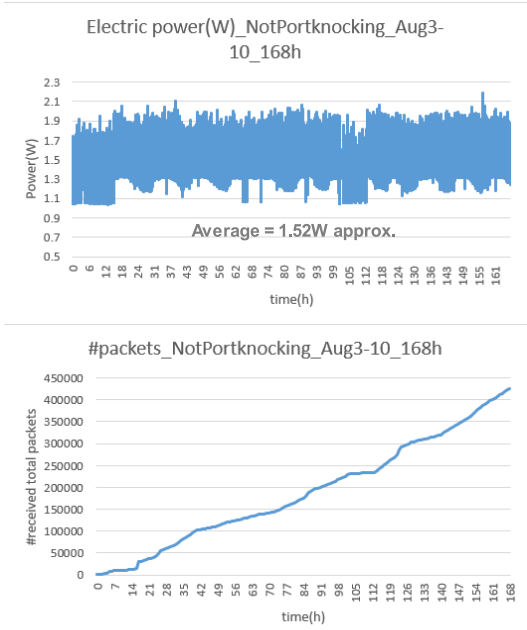


Fig. 12 Power consumption and packets received **without** port knocking during the first week of testing.

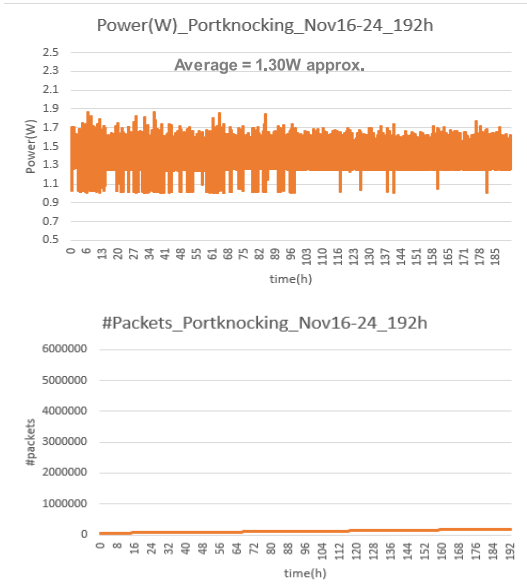


Fig. 13 Power consumption and packets received **with** port knocking during the later week of testing.

on its port 22/TCP was significantly less compared to the device without the port knocking feature. The first week of data collected showed that the IoT device without the port knocking feature had an increase of 0.17 W in its power consumption (1.52 W) from its baseline. Whereas the one with the port knocking feature had only a minimal increase of 0.02 W (1.40 W). The sample of data collected in the first week is shown by the graphs in **Figs. 11** and **12**.

Another data sample collected in later weeks shows the same trend of receiving a very high number of packets on the exposed port compare to a significantly lesser number of packets on the stealthily hidden port. The power consumed by the IoT device without the port knocking was around 1.50 W. Whereas on the one with the port knocking security feature, it was decreased to 1.30 W over the same period of time, as shown in **Figs. 13** and **14**. This shows a difference of approx. 0.20 W between them. The maximum difference observed so far has been an average of 0.25 W.

Hence, the power consumption observed was always less on the IoT device with the stealth port knocking feature than the one with no such feature. The gap between them slowly increases with time as we observed an average of 0.12 W difference in the first week of testing (Figs. 11 and 12), but then it grew to an average of 0.25 W in the later weeks due to the longer exposure of visible SSH default port to the Internet. This indicates that the longer we use the IoT device with a port knocking feature, its power consumption decreases further compared to the one with-

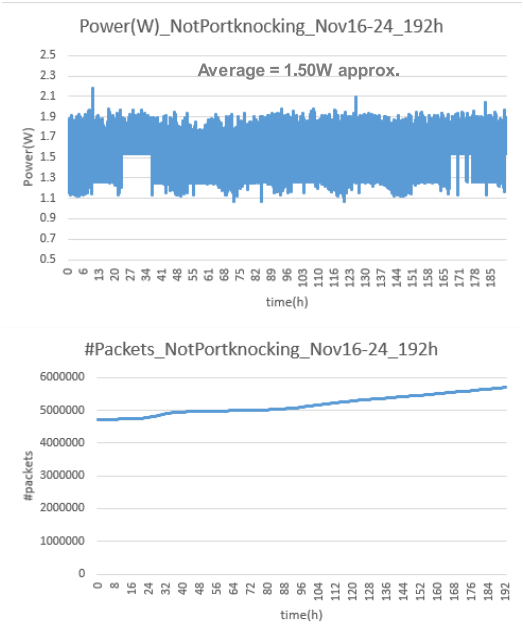


Fig. 14 Power consumption and packets received **without** port knocking during the later week of testing.

out the port knocking feature. The stealthy nature of the port knocking method keeps the default SSH port hidden from the Internet and makes the IoT device not respond to any scans or inquiries for that service port unless the correct knock sequence is received. As a result, the IoT device receives a lesser number of packets than the one with an open service port exposed to the Internet.

4. Conclusion

In the case of port knocking, the stream-cipher-based algorithm was much more practical to use on resource-constrained IoT devices. It not only kept the CPU usage to a very reasonable level and hid the default service port effectively but also helped in greatly reducing the unauthorized traffic coming to that service port. Thereby it helps the resource-constraint IoT device in consuming less power. This could provide a secure remote management option for authorized users without causing much overhead on existing resources of the IoT device. On the other hand, without the stealth port knocking feature, the visible port 22/TCP responded to all the incoming SYN packets that allowed the scanning hosts to follow up with other packets for completing the TCP handshake, enabling attempts to establish or exploit the SSH service. Thereby, received more packets to process and consumed more power than its counterpart that was running the stealth port knocking security feature.

Hence, we can conclude that the experimental results imply that the power consumption overhead by receiving incoming session requests (from scanners/malware on the Internet) would easily exceed the power consumption for running port knocking service. Therefore, running the stealth port knocking feature would be beneficial in terms of not only security enhancement but also power consumption.

5. Considerations

The current study was carried out by testing the port knocking feature only for the SSH service using key-authentication on the default port 22/TCP. We have tested it on the raspberry-pi hardware platform for determining the effectiveness of hiding the port from unauthorized traffic (from scanners/malware on the Internet) and its effect on the IoT device's power consumption, with and without the stealth port knocking security feature. Our scenario is applicable to those IoT devices that are directly connected to the Internet using high-speed cellular networks. Careful considerations were given to ensure we benchmark and calibrate measuring devices before collecting the data to avoid any external influence. Though this proposed solution has shown encouraging results, however, we have not tested it for other services and non-default ports. Also, when considering the defense-in-depth approach, what other security methods can be applied and the choice of security layers must take into account the available resources, as we saw around 50% CPU utilization in the case of the PRNG-CRNG based port knocking solution with our resource-constrained IoT device.

6. Future Works

This study has provided us with some promising results for using the port knocking security concept with which we can provide a secure channel for the remote management of an IoT device using the SSH service without exposing it to unwanted traffic. This method also decreases the total number of packets received by the IoT device on the hidden ports compared to the device without the port knocking feature, which helps in maintaining a lower power consumption.

For future work, this port knocking security feature can also be coupled with other lightweight security options to provide a layered defense (defense-in-depth) approach for the resource-constraint IoT devices. Newly developed port knocking algorithms should be tested to confirm their impact on CPU usage. Expanding the effectiveness of this stealth port knocking feature with other services running on the non-default TCP /UDP ports as well as with multiple services at the same time is recommended to provide more tested options available for the effective use of the port knocking feature. As we have tested the concept of IoT devices being directly exposed to the Internet via the high-speed 3G cellular connection (instead of being behind a router/gateway firewall), therefore, future verification options can include the forthcoming smart cellular IoT devices having a direct 5G high-speed connectivity to the Internet.

Acknowledgments A part of this research work was obtained from an EU-Japan collaboration project "Multi layered Security technologies to ensure hyper-connected smart cities with Blockchain, Big Data, Cloud and IoT (MSEC)", jointly funded by the European Union's Horizon 2020 research and innovation program (contract No.814917) and by the Commissioned Research of National Institute of Information and Communications Technology (NICT), JAPAN (contract No.19501).

References

[1] Symanovich, S.: The future of IoT: 10 predictions about the Internet of Things, Norton Inc. (2019), available from ⟨https://us.norton.com/Internetsecurity-iot-5-predictions-for-the-future-of-iot.html⟩ (accessed 2020-10-21).

[2] Zaidi, A., Branneby, A., Nazari, A., Hogan, M. and Kuhlins, C.: Cellular IoT in the 5G era, Ericsson (2020), available from ⟨https://www.ericsson.com/en/reports-and-papers/white-papers/cellular-iot-in-the-5g-era⟩ (accessed 2020-11-14).

[3] Hung, M.: Leading the IoT, Gartner Inc. (2017), available from ⟨https://www.gartner.com/imagesrv/books/iot/iotEbook_digital.pdf⟩ (accessed 2020-07-14).

[4] Ramakrishna, C., Kumar, G.K., Reddy, A.M. and Ravi, P.: Survey on various IoT attacks and its countermeasures, *International Journal of Engineering Research in Computer Science and Engineering* (*IJERCSE*), Vol.5, No.4, pp.143–150 (2018).

[5] Deogirikar, J. and Vidhate, A.: Security attacks in IoT: A survey, *Proc. International Conference on IoT in Social, Mobile, Analytics and Cloud* (*I-SMAC'17*), pp.32–37 (2017).

[6] Jaramillo, L.E.S.: Malware detection and mitigation techniques: Lessons learned from Mirai DDOS attack, *Journal of Information Systems Engineering & Management*, Vol.3, No.3, Article No.19 (2018), available from ⟨https://doi.org/10.20897/jisem/2655⟩ (accessed 2020-07-14).

[7] Perrone, G., Vecchio, M., Pecori, R. and Giaffreda, R.: The Day after Mirai: A survey on MQTT security solutions after the largest cyber-attack carried out through an army of IoT devices, *Proc. 2nd International Conference on Internet of Things, Big Data and Security* (*IoTBDS'17*), pp.246–253 (2017).

[8] Andreatos, A.S.: Hiding the SSH port via smart Port Knocking, International Journal of Computers, Vol.11, pp.28–31 (2017).

[9] Arifianto, R., Sukarno, P. and Jadied, E.: An SSH honeypot architecture using port knocking and intrusion detection system, *Proc. 6th International Conference on Information and Communication Technology* (*ICoICT'18*), pp.409–415 (2018).

[10] Borges, E.: Top 20 and 200 most scanned ports in the cybersecurity industry (2019), available from ⟨https://securitytrails.com/blog/top-scanned-ports⟩ (accessed 2020-10-21).

[11] Ali, F.H.M., Yunos, R. and Alias, M.A.M.: Simple port knocking method: Against TCP replay attack and port scanning, *Proc. International Conference on Cyber Security, Cyber Warfare and Digital Forensic* (*CyberSec'12*), pp.247–252 (2012).

[12] Vasserman, E.Y., Hopper, N., Laxson, J. and Tyra, J.: SilentKnock: Practical, provably undetectable authentication (2007), available from ⟨https://doi.org/10.1007/978-3-540-74835-9_9⟩ (accessed 2020-08-19).

[13] Khan, Z.A, Javaid, N., Arshad, M.H., Bibi, A. and Qasim, B.: Performance evaluation of widely used port knocking algorithms, *Proc. IEEE 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems* (*HPCC'12*), pp.903–907 (2012).

[14] Sathyadevan, S., Vejesh V., Doss, R. and Pan, L.: Portguard – An authentication tool for securing ports in an IoT gateway, *Proc. IEEE International Conference on Pervasive Computing and Communications Workshops* (*PerCom Workshops'17*), pp.624–629 (2017).

[15] Santis, F.D., Schauer, A. and Sigl, G.: ChaCha20-Poly1305 authenticated encryption for high-speed embedded IoT applications, *Proc. Design, Automation & Test in Europe Conference & Exhibition* (*DATE'17*), pp.692–697 (2017).

[16] Sel, D., Totakura, S.H. and Carle, G.: sKnock: Port knocking for masses, *Proc. IEEE 35th Symposium on Reliable Distributed Systems Workshops* (*SRDSW'16*), pp.1–6 (2016).

[17] Seidel, U.: TCP stealth hides open ports, ADMIN – Network and Security (2015), available from ⟨https://www.admin-magazine.com/Archive/2015/26/TCP-Stealth-hides-open-ports⟩ (accessed 2020-08-19).

[18] deGraaf, R., Aycock, J. and Jacobson, M.: Improved port knocking with strong authentication, *Proc. 21st Annual Computer Security Applications Conference* (*ACSAC'05*), pp.451–462 (2005).

[19] Al-Bahadili, H. and Hadi, A.H.: Network security using hybrid port knocking, *International Journal of Computer Science and Network Security* (*IJCSNS*), Vol.10, No.8, pp.8–11 (2010).

[20] Mehran, P., Reza, E.A. and Laleh, B.: SPKT: Secure port knock-tunneling, an enhanced port security authentication mechanism, *Proc. IEEE Symposium on Computers & Informatics* (*ISCI'12*), pp.145–149 (2012).

[21] Mahbooba, B. and Schukat, M.: Digital certificate-based port knocking for connected embedded systems, *Proc. 28th Irish Signals and Systems Conference* (*ISSC'17*), pp.1–5 (2017).

[22] Yutanto, H.: Security Intelligence for Industry 4.0: Design and implementation, *Theoretical & Applied Science*, Vol.9, No.65, pp.228–243, ISSN 2308-4944 (2018).

[23] ChaCha20 and Poly1305 for IETF Protocols (RFC 7539), available from ⟨https://tools.ietf.org/html/rfc7539⟩ (accessed 2020-06-15).

[24] KDDI Research Inc.: Security analysis of ChaCha20-Poly1305 AEAD, *Cryptography Research and Evaluation Committees*, pp.32–33 (2016), available from ⟨https://www.cryptrec.go.jp/exreport/cryptrec-ex-2601-2016.pdf⟩ (accessed 2020-07-14).

[25] Kereki, F.: Implement port knocking security with Knockd, Linux Journal (2010), available from ⟨https://www.linuxjournal.com/magazine/implement-port-knocking-security-knockd⟩ (accessed 2020-06-24).

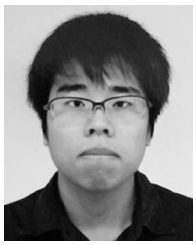
[26] ChaCha20 and Poly1305 for IETF Protocols (RFC 8439), available from ⟨https://tools.ietf.org/html/rfc8439⟩ (accessed 2020-06-22).

[27] Tex2e/ChaCha20-Poly1305 – GitHub, available from ⟨https://github.com/tex2e/chacha20-poly1305⟩ (accessed 2020-06-25).

[28] OpenBlocks IoT VX2: Plat'Home (2019), available from ⟨https://www.plathome.co.jp/product/openblocks-iot/vx2/⟩ (accessed 2020-06-30).

[29] OpenBlocks IoT BX1: Plat'Home (2019), available from ⟨https://www.plathome.co.jp/product/openblocks-iot/bx1/⟩ (accessed 2020-06-30).

Aamir H. Bokhari is pursuing his Ph.D. in Informatics at Yokohama National University. He received his M.Sc. degree in cybersecurity from the University of Dallas, an excellence center of USA's department of homeland security (DHS) and national security agency (NSA). He is also working as the lead researcher on the EU-Japan project of multi-layered security solutions for smart cities (M-Sec). His background and research interests cover computer and network security, IoT security, cybersecurity, cloud security, data and information security, IT and telecommunications. He received the JIP Specially Selected Paper Award in 2020.



Yuta Inoue is a second-year student of the master's degree course at Yokohama National University. He is currently working on the effects of security technology on IoT device performance and evasive malwares. His research interests cover computer and network security, including malware analysis and IoT security.

Seiya Kato is a second-year student of the master's degree course at Yokohama National University. He is currently working on the effects of security technology on IoT device performance and honeypots. His research interests cover computer and network security, including IoT security and analysis of cyber-attacks.

Katsunari Yoshioka has been an Associate Professor at Yokohama National University since 2011. Before that, he was a researcher at the National Institute of Information and Communications Technology, Japan. His research interests cover a wide area of system security and network security including malware analysis and IoT security. He received the commendation for science and technology by the minister of MEXT, Japan in 2009, the award for contribution to Industry-Academia-Government Collaboration by the minister of MIC, Japan in 2016, and the Culture of Information Security Award in 2017.

Tsutomu Matsumoto is a professor of the Faculty of Environment and Information Sciences, Yokohama National University, and directs the Research Unit for Information and Physical Security at the Institute of Advanced Sciences. Prof. Matsumoto also serves as the Director of the Cyber Physical Security Research Center (CPSEC) at the National Institute of Advanced Industrial Science and Technology (AIST). Starting from Cryptography in the early '80s, Prof. Matsumoto has opened up the field of security measuring for logical and physical security mechanisms. He received a Doctor of Engineering degree from the University of Tokyo in 1986. Currently, he is interested in the research and education of Embedded Security Systems such as IoT Devices, Cryptographic Hardware, In-vehicle Networks, Instrumentation and Control Security, Tamper Resistance, Biometrics, Artifact metrics, and Countermeasures against Cyber-Physical Attacks. He serves as the chair of the Japanese National Body for ISO/TC68 (Financial Services) and the Cryptography Research and Evaluation Committees (CRYPTREC) and as an associate member of the Science Council of Japan (SCJ). He was a director of the International Association for Cryptologic Research (IACR) and the chair of the IEICE Technical Committees on Information Security, Biometrics, and Hardware Security. He received the IEICE Achievement Award, the DoCoMo Mobile Science Award, the Culture of Information Security Award, the MEXT Prize for Science and Technology, and the Fuji Sankei Business Eye Award.