Bio-Inspired Framework for Security in IoT Devices

Dion Mariyanayagam[1], Dr. Pancham Shukla[1], and Prof. Bal S. Virdee[1]

[1] Centre for Communications Technology
London Metropolitan University London, United Kingdom

Abstract. One of the many things COVID-19 has taught humanity is that the internet is not just a commodity but a vital service integral to the modern world. As we become ever more connected, there is a growing need to secure data and communication streams. If data is valued, then it should be protected. Unfortunately, some of the least secure devices in modern electronic systems are the Internet of Things (IoT) devices- partly due to their low processing power and always-on functionality.

Polymorphism is the notion of changing one's form. In biological organisms, polymorphic (mutating or changing) viruses trick the natural security mechanisms by changing their unique signatures (e.g. DNA or proteins). In computing, antivirus software systems are adapted to detect and remove constantly changing software viruses. However, polymorphism at the firmware level and over the wireless medium is neither well understood nor explored for IoT devices.

This paper proposes a novel and bio-inspired framework for securing distributed IoT devices often assumed to be working at the intersection of engineering, computing, and cybersecurity domains. The proposed framework attempts to exploit the notion of polymorphism in resource-constrained (e.g. memory, power, bandwidth) IoT devices. The framework's core aim is to detect, reject, and block foreign agents individually or collaboratively and in real-time within a client and server model by changing the access credentials and encryption keys as soon as an unauthorised client is detected. The framework proposed for the bio-inspired framework for security in IoT devices is designed to remain operationally compartmentalised, functionally integrated, and objectively unified.

Keywords: Internet of Things (IoT), Polymorphism, and Bio-Inspired.

1 Introduction

Internet of Things (IoT), which enables a wide variety of embedded devices, sensors and actuators (known as smart things) to interconnect and exchange data, is a promising network scenario for bridging the physical devices and virtual objects in the cyber world. However, such smart devices and sensitive data are vulnerable to security threats. Security is, therefore, the central area of focus for researchers in the field of IoT. Consequently, it is essential to develop cryptography technologies to secure the data from unauthorised access. Moreover, this can be achieved by transforming the data into an unrecognisable and unrelatable form. However, it is not easy to find one straightforward approach that will fit all IoT applications. There are various types of devices connected to an IoT network. Some devices can afford heavyweight and high-security methodologies, but most IoT devices are resource-constrained. They need a security solution that acts fast. Simultaneously, it needs to be simple in its complexity and versatile.

Last but not least important factor is trusted security. In general, the dedicated cryptographic algorithms need to be lightweight in terms of area, memory footprint, power and energy consumption. Therefore, the proposed research is pertinent since security and privacy in the IoT are not fully addressed.

Polymorphism is a very uncommon term and rarely researched topic within the traditional tech, cybersecurity and engineering industries. However, in the computer science discipline, one knows that a polymorphic code can change the initial code, but the code's function produces the same result; for example, 4+4 and 2x4 both have the same results 8.

The nature of changing or mutating forms such as viruses (e.g. malware) that bypass the security system is more familiar to hackers than many security experts [1]. However, these mutating forms are well studied and documented in the human sciences, particularly biology. Thus, the notion of the proposed bio-inspired framework for security closely follows and resembles the biological functions of the immune system. In particular of a human white blood cell due to its ability to detect (intrusion detection), change (changing encryption keys and passwords) and fight (eject from the network) foreign cells inside the human body [2].

1.1 Framework Operations

The design and idea of the bio-inspired framework for IoT security, like many engineering solutions for real-world problems, takes inspiration from the most experienced engineer who is nature her self—in particular, looking at how the immune system (adaptive immune responses and innate immune responses [3][4]) in the human body. For example, how the immune system fights to combat new viruses and infections through vaccines.

This framework's fundamental nature is based on a server and client model whereby the server and client communication are encrypted. The encryption key is in the ledger (similar to blockchain technologies) and shared with authorised clients. Authorised clients are approved depending on their Universal Unique Identifier (UUID); either they are in the approved UUID address list or not. The server has a list of approved UUID addresses, and only authorised clients can connect to the server. Therefore, all authorised clients have the same ledger. This ledger contains two variables as objects which are inside an array. The first object is the Wi-Fi details (password), and the second object includes the encryption key to which the client and server communicate. The Wi-Fi details automatically change if the server detects any foreign clients trying to connect.

When any form of intrusion is detected, for example, an unauthorised client successfully connects to the server, a signal (trigger) is sent between the server and client to successfully apply the change between one polymorphic form to another, for example like the transition from A -> B, where A is the original form and B is the new form.

The signal (trigger) is a message which contains a randomly chosen number that corresponds to an element in the array (ledger) that has the new credential and encryption key to use, therefore resulting in its new polymorphic form. More detailed information for the framework is in chapter three.

1.2 Framework Aims and Objectives

The bio-inspired framework for security in IoT devices needs to:

- Establish a wireless network with encrypted communication between the server and client devices.
- Have clients that read sensor data and send the data to the server successfully.
- Support a lightweight code to maximise the number of resources the device can utilise for sensors.
- Run multiple sensors from one IoT device without the polymorphic framework interfering and burdening the processing power with an ability to invoke the processor's sleep mode.
- Have the ability to detect unapproved clients.
- Have the means to eject unapproved clients.
- Have the function to change the security form by changing the server credentials and encryption key.
- Successfully reconnect all clients back to the server after changing forms and receive data.

2 LITERATURE REVIEW

Designing with a vision for the future helps devices be in service for longer as the device can be upgraded through software and modified by adding additional hardware. The paper "lattice-based cryptography for IoT in a quantum world: are we ready?" [5] shows how current FPGA's and other IoT platforms could implement lattice-based cryptography. In order to implement lattice-based cryptography in an IoT platform, there are a few challenges. One of the challenges is communication bandwidth, as most embedded processors are memory-constrained, therefore only suited for minor security parameters such as IoT applications with limited transmission bandwidth (through Wi-Fi). Security strength often balances performance and security as any brute-force crypt-analytic efforts require more computational resources increasingly for the required search on a block cypher such as AES-128 and other similar cyphers. However, the trade-off between performance and the required security is generally less desirable due to their associated overhead. This paper gives an excellent overview of what IoT devices are lacking. However, it does open a pandora box to a more significant impact on quantum computing and its ability to crack modern security systems with ease.

One of the challenging issues in IoT devices is the tampering of firmware, as it is challenging to detect and recover from the tampered firmware. The paper "ChainVeri: blockchain-based firmware verification system for IoT environment" [6] proposes a new blockchain-based firmware verification system that used a shared palette (ledger) to check the devices to see any firmware tampering have been made. The palette comprises the block header, which encompasses the block hash, block size, block version, previous block hash, time, difficulty, and nonce. The palette also has another module called the verification information, verifying the device model, firmware version, verifier, and identification. Having these functions in the palette allows the blockchain to know the block’s version when the structure of the palette changes. It can also verify the device through a universally unique identifier and check and verify its firmware.

Due to IoT devices and storage’s limited computational power, traditional off-the-shelf solutions are resource-containing on these devices and therefore not recommended for security implementation. The paper “Pseudo-Random Number Generator and Hash Function for Embedded Microprocessors” [7] proposes a solution to overcome this problem by implementing lightweight techniques for efficient Pseudo-Random Number Generator and Hash function to reduce memory consumption and accelerate performance. The reason behind using a pseudo-random number generator is to build and generate harder to break keys and other secret parameters on embedded processors. This paper does propose a relatively efficient way to harden existing security methods such as AES by implementing a pseudo-random number generator; therefore, not too much change needs to occur on the broader system for integration.

“Lightweight Cryptography Algorithms for resource-constrained IoT devices: A Review, Comparison and Research Opportunities” [8] is an excellent paper that provides a holistic view and compares various lightweight cryptography algorithms that are available in the market. The critical common challenges with conventional IoT based cryptography are limited memory (RAM, ROM, and registers), limited computational power, the small surface area of the device, lower battery power (no battery power with RFID tags), and real-time operations. The way this paper compares the different cryptography algorithms is by three characteristics, physical, which is the physical area, memory and battery power. Performance determines the computing power (latency and throughput); lastly, the security characteristic measures minimum security strength in bits, attack models, and side-channel and fault-injection attacks. With these various parameters, the thesis can benefit by paying closer attention to what is valued in this paper’s comparisons of different lightweight cryptography algorithms.

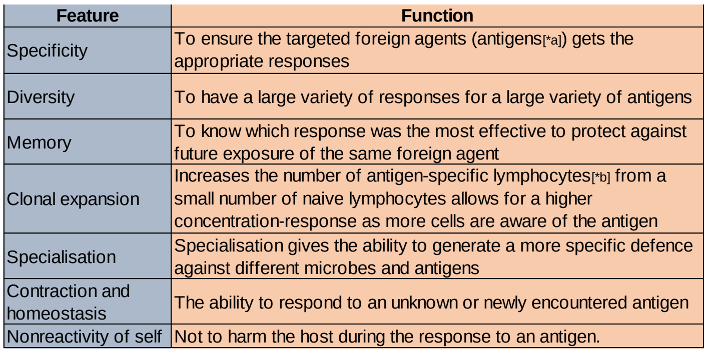
A comprehensive and detailed thought for what type of IoT devices should be implemented in an IoT platform in any IoT system development. Therefore, a study compares and “Reviews Low-End, Middle-End, High-End IoT devices” [9]. The way the IoT devices are classified are as follows: Low-end devices (such as at ATTINY85) are classified as having less than 50kB of RAM, less than 250kB of Flash, devices that do not support an RTOS to devices with RTOS, communication protocols range from gateway communication, lightweight protocols such as Constrained Application Protocol (CoAP), and communication protocols such as HTTP. Lastly, security vulnerabilities whereby data is compromised, causing a medium to a high threat. Middle-End devices (like the ESP8266 and ESP32) provide more outstanding features and processing capabilities such as more RAM, Flash, higher clock speeds, and various communication protocols like Wi-Fi, Bluetooth and Bluetooth Low Energy (BLE). High-end devices (Raspberry Pi, PandaBoard, HummingBoard, and more) typically are single-board computers with powerful processing units and plenty of RAM to provide a graphical user interface or even the ability to run custom operating systems like Windows 10 IoT, Ubuntu, Linux, and Raspberry OS. The main reason why this study is essential for this IoT development is that it helps developers to decide which board to use for a particular functionality. Such as if there is a requirement to gather air pollution data in a given environment, there isn’t a need to use a high-end device as they are more expensive for scaling up, instead of using a middle-end device would prove fruitful as they have enough processing power and a communication protocol to gather the data and send it to a server.

All the related studies have contributed in some way to the development of the framework. In fact, the proposed framework incorporates relevant features from the interdisciplinary research summarised in Section II. For example, the use of a pseudo-random number generator and hash functions is a lightweight approach to develop harder to break keys and is an excellent use case for the trigger system for the proposed framework. Additionally, the labelling and classification of IoT devices into tiers accordingly to their performance and features allows for various testing scenarios on the efficiency of the proposed framework.

3 BIO-INSPIRED (POLYMORPHIC) SECURITY FRAMEWORK

Humans have an innate and adaptive immunity in terms of immunology. Innate immunity responds and recognises generic targets on foreign agents (pathogens), whereas adaptive immunity recognises specific targets using “randomly generated receptors that have a virtually unlimited recognition repertoire” [10]. There are two different types of adaptive immunity called humoral immunity and cell-mediated immunity, and please see the Figure below for a visual representation. The properties of adaptive immune responses are Specificity, Diversity, Memory, Clonal expansion, Specialisation, Contraction and homeostasis, and nonreactivity of self [11]. Please see Table 1 below for each feature and function for adaptive immunity properties.

Table 1. Properties of adaptive immune responses



Feature	Function
Specificity	To ensure the targeted foreign agents (antigens[*a]) gets the appropriate responses
Diversity	To have a large variety of responses for a large variety of antigens
Memory	To know which response was the most effective to protect against future exposure of the same foreign agent
Clonal expansion	Increases the number of antigen-specific lymphocytes[*b] from a small number of naive lymphocytes allows for a higher concentration-response as more cells are aware of the antigen
Specialisation	Specialisation gives the ability to generate a more specific defence against different microbes and antigens
Contraction and homeostasis	The ability to respond to an unknown or newly encountered antigen
Nonreactivity of self	Not to harm the host during the response to an antigen.

[a] An antigen is any toxin or foreign substance that induces an immune response in the body.

[b] B Lymphocytes produce proteins called antibodies which mediates a humoral immune response.

Referring back to human biology, we can see that human immunology can detect foreign agents by using a known threat database, which, if those known threats are detected, an immune response is activated.

For programmers and engineers alike, adaptive immunity seems like an intrusion detection system with a database for known security risks. The detection of a foreign agent that correlates to a known security risk can then be subject to a specifically targeted response. It is also relatively common to block known incoming threats by either whitelisting specific sites, applications or communication streams or blacklisting threats known to the host/organisation. Bio-Inspired framework for security in IoT devices is based around adaptive immunity, and a top-down overview of the framework is shown below in Fig. 1.

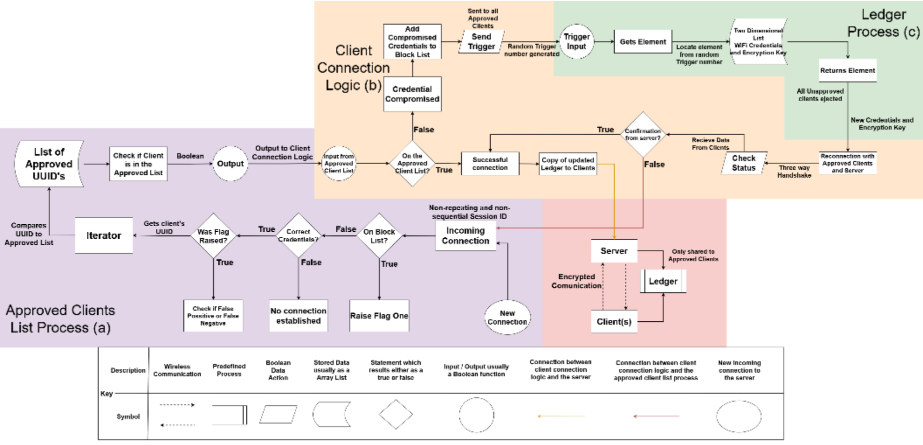

Client Connection Logic (b)
Ledger Process (c)
Approved Clients List Process (a)
List of Approved UUID's
Check if Client is in the Approved List
Boolean
Output
Output to Client Connection Logic
Input from Approved Client List
On the Approved Client List?
True
False
Credential Compromised
Add Compromised Credentials to Block List
Sent to all Approved Clients
Send Trigger
Random Trigger number generated
Trigger Input
Gets Element
Locate element from random Trigger number
Two Dimensional List WiFi Credentials and Encryption Key
Returns Element
All Unapproved clients ejected
New Credentials and Encryption Key
Reconnection with Approved Clients and Server
Three way Handshake
Check Status
Recieve Data From Clients
Confirmation from server?
True
False
Successful connection
Copy of updated Ledger to Clients
Non-repeating and non-sequential Session ID
Incoming Connection
On Block List?
False
True
Correct Credentials?
True
False
Was Flag Raised?
True
Gets client's UUID
Iterator
Compares UUID to Approved List
Check if False Possitive or False Negative
No connection established
Raise Flag One
New Connection
Server
Only shared to Approved Clients
Ledger
Encrypted Comunication
Client(s)
Key
Description
Symbol
Wireless Communication
Predefined Process
Boolean Data Action
Stored Data usually as a Array List
Statement which results either as a true or false
Input / Output usually a Boolean function
Connection between client connection logic and the server
Connection between client connection logic and the approved client list process
New incoming connection to the server

Fig. 1. A high-level Overview of the Bio-inspired Framework for Security in IoT Devices.

3.1 Approved Clients List Process (ACLP)

The beginning of the client journey starts with a new connection. The approved client list is the first line of defence against any unapproved clients. Additionally, the approved client list also acts as a RADIUS Server. The new client connection flows into the incoming connection function, where it is given a session ID that is non-repeating and non-sequential. Once given the session ID, the process then checks if the client has the correct credentials to connect to the server. The framework checks if the incoming client is on the block list and raises a flag if true. If the credentials are incorrect, then there is no connection established between the incoming client and the server. If the credentials are valid, the server will check if the flag for raised to see if there was a false positive or a false negative, then it moves on to request the client's UUID to compare using an iterator (a search function) with the approved UUID addresses. The iterator function will return and output either a “true” or “false” whether the UUID address is on the list or not. Referring back to the properties of adaptive immune responses, the ACLP deeply reflects features such as specificity, diversity, memory and specialisation.

3.2 Client Connection Logic (CCL)

The return output from the approved client list becomes an input variable to the client connection logic, as this function will react accordingly depending on the variable. If the variable is “true” (the connected client is on the UUID address list), this is considered a successful connection, and the server gives a copy of the ledger with the element location for the current password and encryption key. At this stage, the client and the server have successfully connected and can communicate with other clients and the server when necessary. Contrariwise, if the variable returns a “false”, then the implication is that the current server credentials are compromised, and this unapproved client knows the password to connect. As soon as the “false” variable returns, the credentials used are added to the block list, and a trigger (composed from a random number generator) is sent to all approved connected clients. When a client receives the trigger value immediately, it will locate the ledger’s element corresponding to the trigger value. The primary purpose of CCL reflects features of adaptive immunity such as nonreactive to self and contraction and homeostasis.

3.3 Ledger Process (LP)

The ledger only shared between approved clients contains a database comprised of a two-dimensional list, the password to connect to the server (Wi-Fi credentials) and the encryption key required to decypher the communication stream between client to client (node and peer to peer communication) and from client to server. The ledger’s returned element allows the approved clients to reconnect to the server with the new credentials. As a result, all the unapproved clients get automatically rejected from the system as they do not have the newly updated credentials. When the clients attempt reconnection after the trigger is activated, a three-way handshake is performed, the server sends a synchronisation, the client syncs to the server synchronisation and acknowledges it then the server acknowledges the client’s acknowledgement. After a three-way handshake is successful, the client gets a copy of the updated ledger with the credentials’ new element location. However, if the three-way handshake is unsuccessful (the server did not acknowledge the client), then the client is reverted to an incoming connection to double-check if it is in the approved client list and if it is, it will get a copy of the ledger, and if it is not, it will be rejected. LP reflects adaptive immunity functions such as contraction and homeostasis, and clonal expansion.

The server and client’s communication stream is encrypted using a block cypher to add additional security against any form of data theft and intrusion, such as a man in the middle attack listening or probing intrusions.

4 CONCLUSION AND FUTURE WORK

In summation, we presented a novel bio-inspired framework for the security of IoT devices by an amalgamation of interdisciplinary concepts from immunology and computer systems security. Which allows the framework to auto-detect foreign agents, reject unauthorised clients, and the polymorphic ability to change client credentials and encrypted communication streams.

The key features of the proposed framework are auto-detection of foreign agents, rejection of unauthorised clients, and demonstrate the notion of polymorphism by changing client credentials and encrypted communication streams.

Nonetheless, the next step is a system realisation of the bio-inspired framework using different categories of IoT devices. The future works aim to test the encumbrance on the framework for three categories of IoT devices: low-end, middle-end, and high-end. Authors envisage that the effectiveness of the system realisation of this framework will be governed by technological challenges such as onboard memory, processing speed, power consumption/management and fault-tolerant connectivity.

5 ACKNOWLEDGEMENT

This work is supported by the University's Rescaling Funds awarded to Dr Pancham Shukla for developing a PhD researcher. The first author, Dion, would like to thank his PhD supervisor Dr Pancham Shukla and Prof Bal Virdee for their guidance and support.

References

1. X. Li, P. K. K. Loh, and F. Tan, "Mechanisms of polymorphic and metamorphic viruses," Proc. - 2011 Eur. Intell. Secur. Informatics Conf. EISIC 2011, pp. 149–154, 2011, doi: 10.1109/EISIC.2011.77.
2. A. K. Abbas, A. H. Lichtman, and S. Pillai, Basic Immunology: Functions and of the Immune System. 2016.
3. C. A. Janeway, P. Travers, M. Walport, and M. Shlomchik, Immunobiology - The Immune System in Health and Disease, 5th ed. New York: Garland Publishing Library, 2007.
4. K. Murphy, C. Weaver, M. & Weaver, R. Geha, and L. Notarangelo, Janeway's Immunobiology, 9th ed. New York and London: Garland Science, 2017.
5. A. Khalid, S. McCarthy, M. O'Neill, and W. Liu, "Lattice-based Cryptography for IoT in A Quantum World: Are We Ready?," Proc. - 2019 8th Int. Work. Adv. Sensors Interfaces, IWASI 2019, pp. 194–199, 2019, doi: 10.1109/IWASI.2019.8791343.
6. J. Lim, Y. Kim, and C. Yoo, "ChainVeri : Blockchain-based Firmware Verification System for IoT environment," pp. 1050–1056, 2018, doi: 10.1109/Cybermatics.
7. H. Seo, J. Choi, H. Kem, T. Park, and H. Kim, "Pseudo random number generator and Hash function for embedded microprocessors," 2014 IEEE World Forum Internet Things, WF-IoT 2014, pp. 37–40, 2014, doi: 10.1109/WF-IoT.2014.6803113.
8. V. A. Thakor, M. A. Razzaque, and M. R. A. Khandaker, "Lightweight Cryptography Algorithms for resource-constrained IoT devices: A Review, Comparison and Research Opportunities," IEEE Access, pp. 1–17, 2021, doi: 10.1109/ACCESS.2021.3052867.
9. M. O. Ojo, S. Giordano, G. Procissi, and I. N. Seitanidis, "A Review of Low-End, Middle-End, and High-End Iot Devices," IEEE Access, vol. 6, pp. 70528–70554, 2018, doi: 10.1109/ACCESS.2018.2879615.
10. D. Flower and J. Timmis, In Silico Immunology, 1st ed., vol. 1, no. 1. London: Springer, 2006.
11. A. K. Abbas, A. H. Lichtman, and S. Pillai, Basic Immunology: Functions and of the Immune System. 2016.