IOT SECURITY ANALYSIS USING NEURAL KEY EXCHANGE PROTOCOL

Lela Mirtskhulava
Iv. Javakhishvili Tbilisi State University, 3 University Str. Tbilisi, Georgia

Nana Gulua
Sokhumi State University, 9 Jiqia Str. Tbilisi, Georgia

Nugzar Meshveliani
Sokhumi State University, 9 Jiqia Str. Tbilisi, Georgia

Abstract
Securing IoT (the Internet of Things) is critical issue concerning to data integrity and the resources within enterprises. IoT security requires a systematic approach for monitoring all possible threats and the methods to mitigate them. Encryption is main requirement for securing IoT through secure communication. Key exchange plays a crucial role in securing an information exchanging through IoT network. Neural networks provide great strategy by synchronization process using Hebbian learning rule by balancing weights. Neural networks synchronization gives us a cryptographic key-exchange protocol. Main benefit of this process is that an attacker needs so long time to guess the generated key.

Keywords: *IoT security, Neural key exchange, Encryption, Tree Parity Machine, Neural Network*

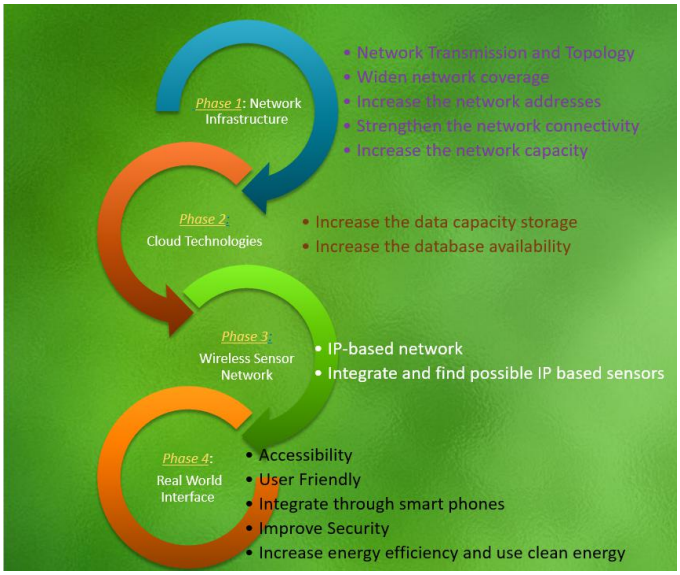
1. Introduction

IoT devices are prone to threats since they are managed by humans. The attacker may try to theft confidential information by gaining unauthorized access to IoT devices. Prior to protect our system or network against the threats first we need to identify the vulnerabilities and the threats caused by Internet connection. Intruders may have the different purposes like to gain illegitimate access to IoT device and therefore to gain confidential information [1-5]. Due to low power and less computing opportunity, IoT devices cannot use complex protocols what gives intruders capable having IoT as easy target. There are hardware and software vulnerabilities in IoT Devices. Most challenging issue is a hardware vulnerability what is too hard to detect but harder to repair the damaged part of the device. Poor algorithm causes software vulnerability providing a back door to intruders to spy [6-8].

We may differentiate two types of threats to IoT: a natural and human threats. The threat that may occur due to hurricanes or earthquakes can damage IoT devices and impossible to repair. Human threats we need to localize are malicious attacks. There are main Attacks on IoT Devices: cyber-attacks where intruder cracks encryption to obtain the keys and malicious software to gain secret information; brute force attacks where intruder makes plenty of attempts using scanning software to guess a password of specific user and third one is tracking where intruder captures victim's move using IoT device UID (Unique Identifier) [9-10].

The common vision of smart systems like smart grid, smart homes, smart water networks, intelligent transportation is usually associated with the concept of the internet of things (IoT), where through the use of sensors the entire physical infrastructure is closely coupled with information and communication technologies.

Intelligent monitoring and management can be achieved using networked embedded devices where devices are interconnected to transmit useful measurement information and control instructions via distributed sensor networks [11-13].



Pic. 1. Four Phases of Internet of things (IoT)

2. Major Human Attacks to IoT

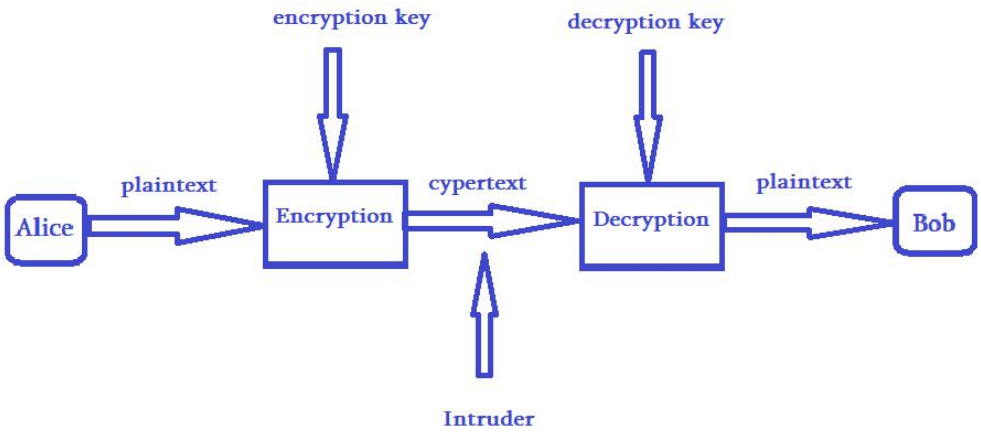
Brute Force Attack: the attackers try to guess a password using specific software making multiple attempts for gaining access.

Cyber Reconnaissance: the attacker trying to use malicious software and cracking technique for spying to obtain about targeted user.

Tracking: each action of the targeted user is fixed using UID of IoT device. Tracking gives a precise location of the targeted user.

3. General Cryptography Model

We know three general encryption forms: Symmetric key encryption, public Key encryption and Cryptographic hash. Symmetric key encryption technique generates identical encryption and decryption keys. The forms symmetric key encryption are AES, DES, 3DES and RC5. In public Key or asymmetric encryption, encryption key is generated publicly and it could be used by anyone for encrypting data but only the receiver having the private key can decrypt the message. Asymmetric cryptography can control data security, authentication of participants. Asymmetric encryption protocols are RSA, Elliptic Curve, TLS PGP and S/MIME.



Pic. 2. General model of symmetric Cryptography

4. Neural Cryptography

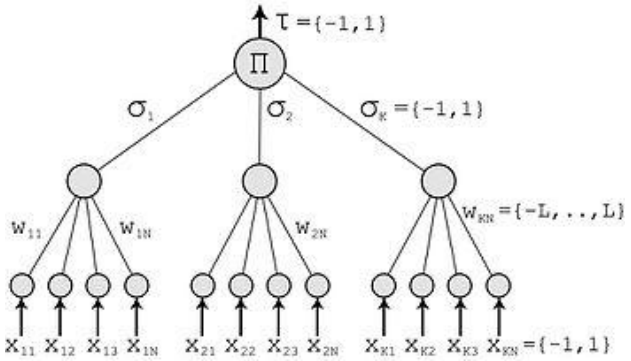
Neural cryptography, the branch of cryptography, is capable to analyse ANN (artificial neural network) algorithms in encryption process and cryptanalysis. ANNs are known as the best solution in cryptanalysis for attack ciphering algorithms to find the inverse function of the cryptographic algorithms. The ability of learning and self-learning as well as stochastic behavior of neural networks is used in public-key cryptography. Mutual synchronization of ANNs, can solve the key distribution problem generating pseudo-random numbers. In cryptanalysis main feature of ANNs is they are capable to separate space in non-linear parts using bias [14-17].

An artificial neural network (ANN) is a math structure which can identify a nonlinear relationships between input and output data sets.

$$a = f(W_p + b)$$

5. Neural Key Exchange Protocol

Key exchange protocol between two parties is Diffie-Hellman protocol well known in practice. For most security reasons we will use neural key exchange protocol using the synchronization of two tree parity machines [18-19].



Pic. 3. Tree Parity Machine

We use feedforward neural network as the tree parity machine with one output layer, k hidden layers and K×N input layers. Where output of each hidden neuron is sum of all multiplications of input neurons and weights. Inputs and Outputs are binary.

Input values: $x_{ij} \in \{-1, 0, +1\}$

Weights: $w_{ij} \in \{-L, \ldots, 0, \ldots, +L\}$

Output of hidden layers is a sum of multiplications of input values and weights:

$$\sigma_i = sgn(\sum_{j=1}^{N} w_{ij} x_{ij})$$

Signum: $sgn(x) = \begin{cases} -1 & if\ x < 1 \\ 0 & if\ x = 1 \\ +1 & if\ x > 1 \end{cases}$

Output value: $\tau = \prod_{i=1}^{K} \sigma_i$

What are main steps of key Exchange protocol: both A and B parties use their own tree parity machines and synchronization of them can be achieved by doing the following steps by: 1) Initializing random weights; 2) executing all these steps until the synchronization is achieved; 3) Generating input vector X; 4) Computing the values of hidden layers; 6) Computing output value; 7) Comparing three parity machines values; 8) If outputs are different, go to 2.1; 9) If outputs are the same, apply learning rules to weights.

When weights of both tree parity machines are equal what happens after completion of synchronization, they will be used as keys by both sides. This is kind of bidirectional learning.

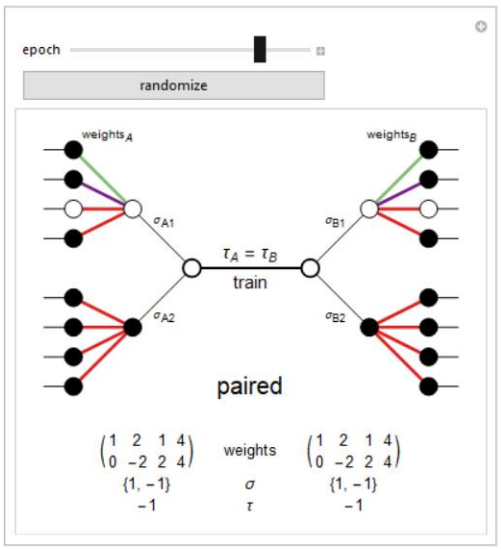
There are three rules can be used for synchronization:

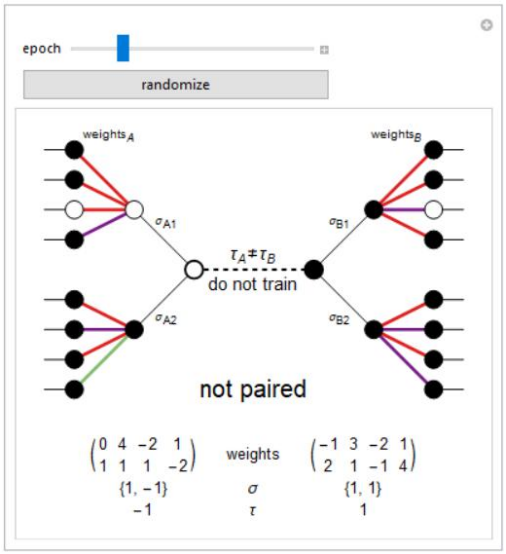
Hebbian learning rule: $w_i^+ = w_i + \delta_i x_i \theta(\delta_i \tau)\theta(\tau^A \tau^B)$
Anti-Hebbian learning rule: $w_i^+ = w_i - \delta_i x_i \theta(\delta_i \tau)\theta(\tau^A \tau^B)$
Random Walk: $w_i^+ = w_i + x_i \theta(\delta_i \tau)\theta(\tau^A \tau^B)$

6. Wolfram Key Exchange Application

This application gives us an opportunity to use a key exchange protocol through the synchronization of two neural networks for encrypting communication using the Hebbian learning rule. The given model includes two parties A and B where the person A communicates with the person B. They need to exchange a key through a secure channel what is impossible until both parties set absolutely identical neural networks and the weights and inputs of both networks match. Changing epoch value can change system using randomizing button creating new neural network configuration. Number of epochs taken to get paired networks was equal to 1000 what was achieved when the weights of the both neural networks matched.

In the given algorithm, A and B parties represent two similar neural networks with different random values of weights: $w_{ij} \in \{-L, \ldots, 0, \ldots, +L\}$ where L is the number of weight values. The input values of the network are random: $x_{ij} \in \{-1, +1\}$. The values of hidden layers is computed using the formula: $\sigma_i = sgn(\sum_{j=1}^{N} w_{ij} x_{ij})$. Output value $\tau = \prod_{i=1}^{K} \sigma_i$. After comparing the output values of both parties and if they do match the Hebbian learning rule will be used where the process is repeated until the weights of both neural networks get equal. These values of the weights gives us the paired key.





Pic.1. Synchronization of Tree Parity Machines (Epoch <1000)

Pic.2. Synchronization of Tree Parity Machines (Epoch >1000)

7. Conclusion

Neural key exchange protocol is good alternative for communication encryption using the Hebbian learning rule. Where part A needs to establish secure communication with part B through the secure channel. They need to exchange a key through this secure channel and they use two neural networks with identical topology. These neural networks are evaluated with the same inputs until their weights match. Neural networks were trained with the number of epochs equal to 1000.

References

[1] [11] J. Hou, L. Qu, and W. Shi, "A survey on internet of things security from data perspectives," Computer Networks, vol. 148, pp. 295 – 306, 2019.

[2] [10] K. A. da Costa, J. P. Papa, C. O. Lisboa, R. Munoz, and V. H. C.

[3] de Albuquerque, "Internet of things: A survey on machine learningbased intrusion detection approaches," Computer Networks, vol. 151, pp. 147 – 157, 2019.

[4] M. Ammar, G. Russello, and B. Crispo, "Internet of things: A survey

[5] on the security of iot frameworks," Journal of Information Security and Applications, vol. 38, pp. 8 – 27, 2018.

[6] [8] I. Stellios, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, and J. Lopez, "A survey of iot-enabled cyberattacks: Assessing attack paths to critical infrastructures and services," IEEE Communications Surveys Tutorials, vol. 20, pp. 3453–3495, Fourthquarter 2018.

[7] [9] F. Restuccia, S. DOro, and T. Melodia, "Securing the internet of things in the age of machine learning and software-defined networking," IEEE Internet of Things Journal, vol. 5, pp. 4829–4842, Dec 2018.

[8] [12] D. E. Kouicem, A. Bouabdallah, and H. Lakhlef, "Internet of things security: A top-down survey," Computer Networks, vol. 141, pp. 199– 221, 2018.

[9] [13] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, "Internet of things security: A survey," Journal of Network and Computer Applications, vol. 88, pp. 10 – 28, 2017

[10] A. Mosenia and N. K. Jha, "A comprehensive study of security of internet-of-things," IEEE Transactions on Emerging Topics in Computing, vol. 5, pp. 586–602, Oct 2017.

[11] An Internet of Things: Reference Architecture. Symantec. 2016

[12] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," IEEE Communications Surveys Tutorials, vol. 17, pp. 2347–2376, Fourthquarter 2015.

[13] ITU Internet Reports 2005: The Internet of Things. Executive Summary. International Telecommunication Union (ITU), Geneva. ITU, 2005.

[14] A. Argyris, D. Syvridis, L. Larger, V. Annovazzi-Lodi, P. Colet, I. Fischer,J. Garc´ıa-Ojalvo, C. R. Mirasso, L. Pesquera, and K. A. Shore. Chaos-based communications at high bit rates using commercial fibre-optic links. Nature, 437(7066):343–346, 2005

[15] Kanter, I. & W.Kinzel, "The Theory of Neural Networks and Cryptography", Minerva Center, Bar-Ilan Uni, Israel, 2003.

[16] Arecchi, F. Tito, "Chaotic neuron dynamics, synchronization, and feature binding: Quantum aspects", Australian National Uni., 2003.
I. Kanter, W. Kinzel, and E. Kanter. Secure exchange of information by synchronization of neural net. Europhys. ., 57(1):141–147, 2002.

[17] W. Kinzel and I. Kanter. Interacting neural networks and cryptography. In B. Kramer, editor, Advances in Solid State Physics, volume 42, pps 383–391. Springer, Berlin, 2002.

[18] W. Kinzel, R. Metzler, and I. Kanter. Dynamics of interacting neural networks. J. Phys. A: Math. Gen., 33(14):L141–L147, 2000.

[19] R. Metzler, W. Kinzel, and I. Kanter. Interacting neural networks. Phys. Rev. E, 62(2):2555–2565, 2000.

J. Hertz, A. Krogh, and R. G. Palmer. Introduction to the Theory of Neural Computation. Addison-Wesley, Redwood City, 1991.