International Journal on Future Revolution in Computer Science & Communication Engineering
Volume: 3 Issue: 1
ISSN: 2454-4248
04 – 08

GIVE-AND-TAKE KEY PROCESSING for Cloud- linked IoT

Dr. M. Sughasiny
Assistant Professor, Department of Computer Science
Srimad Andavan Arts and Science College (Autonomous)
Trichy, Tamilnadu, Inida
Sughasiny5.cs@gmail.com

Abstract—Internet of Things (IoT) is estimated there are over a billion internet users and rapidly increasing. But there are more things on the internet than there are people on the internet. This is what it has been generally mean, when it has been say internet of things. There are millions and millions of devices with sensors that are linked up together using networks that generate a sea of data. The problem is all data needs to remain secured, unchanged, and persisted at each stage of an IoT solution. This includes distributed components, communication infrastructure, back-end analytics and database servers, across potentially remote locations and adverse environments.

In any case, it is helpless against eavesdropping which represents a risk to privacy and security of the client. The security of data traffic winds up plainly vital since the communications over open network happen frequently. It is along these lines basic that the data traffic over the system is encrypted. To give the QoS, the Cloud- linked IoT security is the essential part of the service providers. This paper is concentrating on issue identifying with the Cloud- linked IoT security in virtual condition. It has been propose a technique GIVE-AND-TAKE KEY PROCESSING for giving data process and security in Cloud- linked IoT using Elliptical Curve Cryptography ECC and Hash Map. Encourage, depicts the security services incorporates generation of key, encryption and decryption in virtual condition.

Keywords-*IoT; Cloud; SaaS; ECC; QoS; Encryption; Decryption; Hash Map.*

*****

I. INTRODUCTION

Internet of Things (IoT) [1] is a multidisciplinary domain that covers a large number of topics from purely technical issues (e.g., routing protocols, semantic queries), to a mix of technical and societal issues (security, privacy, usability), as well as social and business themes. IoT applications, both existing and potential, are equally diverse. If it has been are to summarize all of them into one, it is enabling the machine perception of the real world and seamless interactions with it. Environmental and personal health monitoring, monitoring and control of industrial processes including agriculture, smart spaces, and smart cities are just some of the examples of IoT applications.

Internet of Things frameworks might help support the interaction between "things" and allow for more complex structures like Distributed computing [2] and the development of Distributed applications. Currently, Internet of Things frameworks seem to focus on real time data logging solutions offering some basis to work with many "things" and have them interact. Future developments might lead to specific Software development environments to create the software to work with the hardware used in the Internet of Things.

A platform is anything that can run applications and store data. In an organization's data centres, for example, you might have computers running Windows Server and other software that provide a platform for your in-house applications. A cloud platform is the same thing: It's a foundation for running applications and storing data. The biggest difference is that it runs in data centres owned by an external service provider, such as Microsoft, Amazon, Google, IBM, Rack Space etc., and it's accessed via the Internet. Another difference is rather than rely on the exact same technology that your current in-house platform uses, cloud platforms often use something slightly different. For example, Microsoft's cloud platform runs Windows Azure rather than Windows Server.

II. CLOUD PLATFORM

Cloud Platform is a cloud computing service by Cloud Service Providers (CSP) [3] that offers hosting on the same supporting infrastructure that CSP uses internally for end-user products like Cloud based Search engine. Cloud Platform provides developer products to build a range of programs from simple websites to complex applications.

A. So what are the benefits of using the cloud platform? [4]

- Faster deployment of new business capabilities
- Lower-risk business innovation
- Global scale and global reach
- More intelligent IT spending

B. What are the risks of using the cloud platform?

- Outsourcing to an external provider
- Storing data outside your organization (This is clearly a risk but how much risk is appropriate or acceptable is a business decision.)
- Vendor lock-in (so far there is no easy way to port an application developed for one cloud provider to another cloud provider.)
- Major risks is Security (Cloud- linked IoT Security)

III. ELLIPTIC CURVES CRYPTOGRAPHY

ECC is an asymmetric cryptography algorithm [5] which involves some high level calculation using mathematical curves to encrypt and decrypt data. It is similar to RSA as it's asymmetric but it uses a very small length key as compared to

 ISSN: 2454-4248
04 – 08

RSA. ECC is an asymmetric cryptography algorithm which involves the following steps:

A. ECC Encryption

1) Define a Curve
2) Generate public private Key pair using that curve, for both sender and receiver
3) Generate a Shared secret key from the key pair
4) From that shared secret key, generate an encryption key
5) Using that encryption key and symmetric encryption algorithm, encrypt the data to send.

B. ECC Decryption

The sender will either share the curve with receiver or sender and receiver will have the same use for the same curve type. Also, sender will share its public key with receiver.

1) Generate public private Key pair using the same curve for that curve. for receiver.
2) Regenerate a shared secret key using private key of receiver and public key of sender.
3) From that shared secret key, generate an encryption key
4) Using that encryption key and symmetric encryption algorithm, decrypt the data.

TABLE1: Comparison of Major Public Key Encryption Algorithms

	RSA	ELGAMAL (DIFFIE-HELLMAN)	ECC
Patent status	Expired in 2001	Expired in 1997	Not patented, but many low level optimizations are
Mathematical problem	Integer factorization	Discrete logarithm over a finite field	Discrete logarithm of elliptic curve
Best known attack	Subexponential	Exponential	Exponential
Average key size	1024 bits	1024 bits	160 bits
Corporate "champion"	RSA Security	None	Certicom

IV. CLOUD- LINKED IOT ATTACKS

Internet of things (IoT) technologies are getting ready to transform the way it has been work and live; if one thing can prevent the IoT from transforming the way it has been live and work, it will be a breakdown in security. The goal of the paper is to integrate and evaluate the standardized technique GIVE-AND-TAKE KEY PROCESSING Cloud- linked IoT using Elliptical Curve Cryptography (ECC) in modern Internet of Things architectures consisting of things (sensor nodes), clouds, smartphones, and end-users.

A. Physical Attacks

These types of attacks tamper with the hardware components and are relatively harder to perform because it requires expensive material. Some examples are de-packaging of chip, layout reconstruction, micro-probing, and particle beam techniques [6].

B. Side Channel attacks

These attacks are based on "side channel Information" that can be retrieved from the encryption device that is neither the plaintext to be encrypted nor the cipher text resulting from the encryption process. Encryption devices produce timing information that is easily measurable, radiation of various sorts, power consumption statistics, and more. Side channel attacks makes use of some or all of this information to recover the key the device is using. It is based on the fact that logic operations have physical characteristics that depend on the input data. Examples of side channel information are timing attacks, power analysis attacks, fault analysis attacks, electromagnetic attacks, and environmental attacks [7].

C. Cryptanalysis attacks

These attacks are focused on the cipher text and they try to break the encryption, i.e. find the encryption key to obtain the plaintext. Examples of cryptanalysis attacks include Cipher text-only attack, Known-plaintext attack, Chosen-plaintext attack, Man-in-the-middle attack, etc [8].

D. Software Attacks

Software Attacks are the major source of security vulnerabilities in any system. Software attacks exploit implementation vulnerabilities in the system through its own communication interface. This kind of attack includes exploiting buffer overflows and using Trojan horse programs, worms or viruses to deliberately inject malicious code into the system [9].

E. Network Attacks

Wireless communications systems are vulnerable to network security attacks due to the broadcast nature of the transmission medium. Basically attacks are classified as active and passive attacks. Examples of Passive attacks include Monitor and Eavesdropping, Traffic Analysis, Camouflage Adversaries, etc. Examples of Active attacks include Denial of Service Attacks, Node Subversion, Node Malfunction, Node Capture, Node Outage, Message Corruption, False Node, Routing Attacks, etc [10].

F. Major Security concerns for IOT

a) Secure network access: This provides a network connection or service access only if the device is authorized.

b) Secure data communication: It includes authenticating communicating peers, ensuring confidentiality and integrity of communicated data, preventing repudiation of a communication transaction, and protecting the identity of communicating entities.

 ISSN: 2454-4248
04 – 08

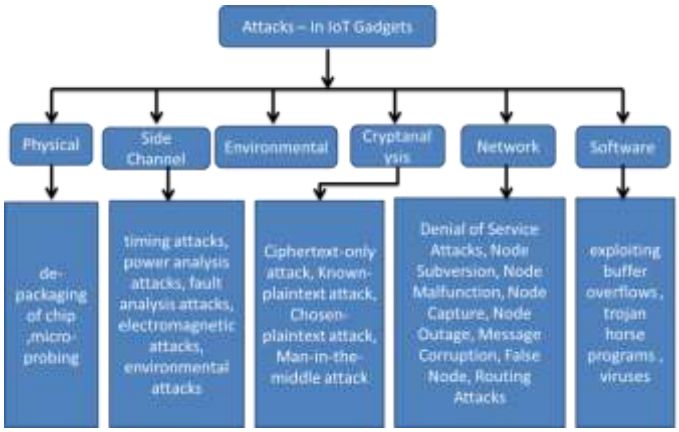

FIGURE: 1: Cloud- linked IoT Attacks

V. PROPOSED GIVE-AND-TAKE KEY PROCESSING FOR CLOUD- LINKED IOT SECURITY FRAMEWORK

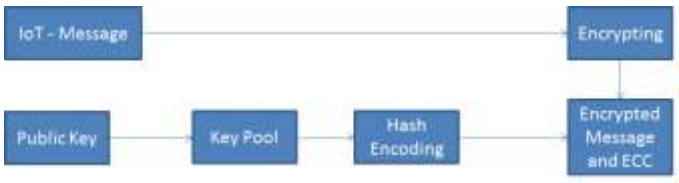

FIGURE 2: Encryption phase

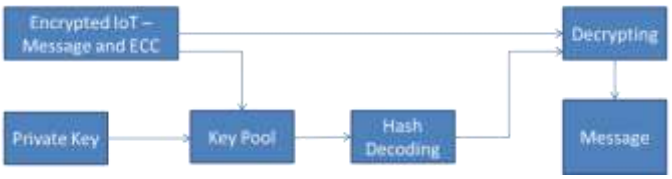

FIGURE 3: Decryption phase

A. Encryption and Decryption -Give-And-Take Key Processing Algorithm

The Elliptic Curve encryption with Hash Map (ECCH) algorithm does have a fairly complicated set of domain parameters that describe what curve is being used. However, after the parameters are chosen (either generated or selected from a table), the actual algorithm for encryption is very simple. The domain conditions are as follows:

a) PARAMETERS

1. The field, either p, a large prime or 2m. These values can be predetermined and preselected and may be reused. The size of p or m needs to be sufficiently large in order in order allow Condition 3.

2. Curve coefficients a, b that are elements of the field that define an elliptic curve E.

3. A prime number r that divides the number of points on E.

4. A curve point G of order r. The size of r (in bits) is what determines the strength of the EC family; normally the minimum is 161 bits.

5. If a binary field is used, a parameter specifying the coordinate representation.

6. Optionally, a cofactor k such that k = o(E)/r.

7. Additionally, in some cases, GCD(k, r) = 1.

The generation for prime fields isn't so bad, but the binary case requires using various tables to aid in computation. Fortunately, the domain parameters can be reused. If that sounds odd, think of being able to "reuse" integers even though you generated an RSA key pair. However, unlike RSA key pairs, these parameters will need validation.

Using our previous example, it has been use the E(field = F11, a = 1, b = 1, o(E) = 14). The point G = (6, 5) has order 7, since 7 × (6, 5) = O. And, finally, k = 14/7 = 2, and GCD(7, 2) = 1.

b) KEY GENERATION

Once you have the all the domain parameters (either generated or validated), generating keys is very simple:

1. Pick a random integer 0 < s < r.

2. Compute a point on the curve W = sG.

W is the public key, and s is the private key. Key generation is fast once the domain parameters are computed. There are other algorithms for validating public keys to make sure points aren't completely bogus. Using our example again, we'll pick s = 4, so W = 4G = 4 (6, 5) = (3, 3).

c) KEY AGREEMENTS AND ENCRYPTION

Just as in the Diffie-Hellman case, it has been can define key agreements. Regardless of the representation or field, the process is identical. Given two sets of key pairs (W1, s1) and (W2, s2), the shared secret value is computed by using:

s2W1 = s1W2

The shared secret is a point on the curve, and the coordinates are effectively random. However it's traditional to use just the x-coordinate. The bits that make up the x-coordinate can be used as a secret key for a symmetric cipher. Continuing our example using a key of ((3, 3), 4); we'll pick another key ((0, 10), 5) and compute the secret.

5 (3, 3) = (6, 6)

4 (0, 10) = (6, 6)

d) HASH FUNCTION

K = key

M = message

H = cryptographic hash function

HMAC = H(k || m)

A more secure approach is the so-called padded envelope, where you make sure the internal hash algorithm is iterated more than once by adding padding after the first key:

HMAC = H(k || p || m || k)

Now that it has been have a secret value, encryption and decryption work identically to the regular discrete logarithm version.

VI. EXPERIMENTAL RESULTS AND ANALYSIS

Mapping the dependencies between the parameter setting and the efficiency features.

Tests were made with Proposed Process, ECC and with RSA during the

- Key Generation
- Establishment Common Parameters
- Encryption
- Decryption
- Signing
- Signature Verification

a) THE MEASURED DATA WERE

– The time of execution

– and the size of data, namely:

 ISSN: 2454-4248
04 – 08

- Size of the Common Parameter Files,
- Size of the Public and Secret Key Files,
- Size of the Encrypted Data Files,
- Size of the Signature Files

b) THE PARAMETERS OF THE OPERATIONS ARE:

- the size of the applied key
- the size and content of the input data

Our experiments used the Apache 2.0.45 web server compiled with OpenSSL-SNAP-20030309 using the Sun Forte Developer 7 C compiler without architecture-specific optimizations. This snapshot of the development version of OpenSSL includes ECC code contributed by Sun Microsystems Laboratories [29]. Enhancements were made to the mod ssl component of Apache in order to make it ECC aware. It has been ran the server on a single 900 MHz UltraSPARC III processor with 2GB of memory inside a Sun Fire V480 server running the Ubuntu 16.4 operating system.5 For the HTTPS clients, it has been used a prototype Sun Fire server equipped with seven 900 MHz UltraSPARC III processors, 14GB of memory and also running the Solaris 9 operating system. The server and client machines were connected via a 100Mb Ethernet network.

B. Analysis of Experimental Results - Comparison of ECC and RSA micro benchmarks

Table 2 shows a comparison of the RSA and ECCH cryptographic operations performed by an Cloud-IoT server. It has been used the OpenSSL speed program to measure RSA decryption and ECDH operation for different key sizes (a minor enhancement was made for collecting RSA-1536 numbers). These micro-benchmarks highlight ECCH's performance advantage over RSA for different security levels. Note how ECCH's performance advantage increases even faster than its key-size advantage as security needs increase.

TABLE 2: Measured performance of public-key algorithms.

	ECCH-160	RSA-1024	ECCH-192	RSA-1536	ECCH-224	RSA-2048
Ops/sec	271.3	114.3	268.5	36.4	195.5	17.8
Performance ratio	271.3		7.4 : 1		21.4 : 1	
Key-size ratio	2.4 : 1		1:08		01:09.1	

C. Relative Costs in an HTTPS Fetch

Figure 4 shows the average time taken by the server to fulfill an HTTPS request for different page sizes and public keys with no session reuse. It has been used micro benchmark results for ECCH, RSA, RC4, and SHA to estimate the relative costs involved. RSA decryption continues to be the dominant cost in all of these cases. According to SPECWeb99 which models real-world web traffic, 85% of the files are under 10KB. For such files, RSA takes up anywhere between 63% to 88% of the overall time depending on the security level. This suggests that efforts to reduce the RSA cost or replace it with a cheaper alternative will have a significant payoff. Indeed, it has been see that using ECCH reduces overall processing time at the server by 29% to 86% across the entire range of page sizes in our study

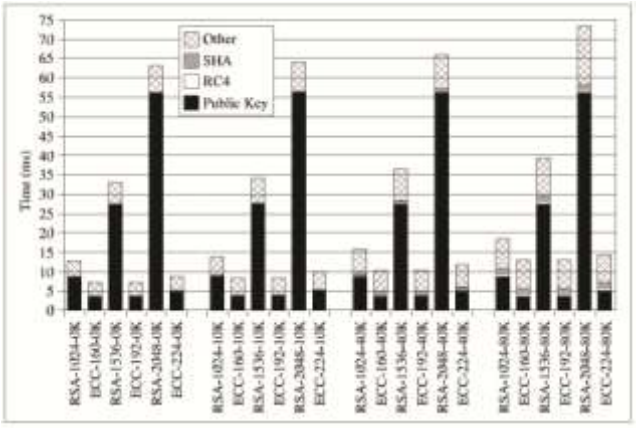

FIGURE 4: Relative costs in an data transaction.

VII. CONCLUSION

The above analysis suggests that the use of ECCH cipher suites offers significant performance benefits to IoT clients and servers especially as security needs increase. Already, there is considerable momentum behind widespread adoption of the Advanced Encryption Standard (AES) and RSA which specifies the use of 128-bit, 192-bit and 256-bit symmetric keys. As indicated in Table 2, key sizes for public-key cryptosystems used to establish AES and RSA keys will correspondingly need to increase from current levels. Furthermore, as users become increasingly sensitive to on-line privacy issues, they are likely to demand IoT protection for more of their transactions. It has been believe these trends bode well for broader deployment of ECCH, in not just wireless environments but also traditional servers/cloud server environments. Besides Cloud-IoT, It has been also added ECCH support. It has been now targeting ECCH support in these servers and intends to perform a similar study for their representative workloads.

REFERENCES

[1] Ahlmeyer, Matthew, and Alina M. Chircu. "SECURING THE INTERNET OF THINGS: A REVIEW." *Issues in Information Systems* 17.4 (2016).

[2] Botta, Alessio, et al. "Integration of cloud computing and internet of things: a survey." *Future Generation Computer Systems* 56 (2016): 684-700.

[3] Porambage, Pawani, et al. "The quest for privacy in the internet of things." *IEEE Cloud Computing* 3.2 (2016): 36-45.

[4] Girau, Roberto, Salvatore Martis, and Luigi Atzori. "A cloud-based platform of the social internet of things." *Internet of Things. IoT Infrastructures: Second International Summit, IoT 360° 2015, Rome, Italy, October 27-29, 2015. Revised Selected Papers, Part I*. Springer International Publishing, 2016.

[5] Liu, Zhe, et al. "Elliptic curve cryptography with efficiently computable endomorphisms and its hardware implementations for the internet of things." *IEEE Transactions on Computers* 66.5 (2017): 773-785.

[6] Pan, Yao, et al. "Taxonomies for Reasoning About Cyber-physical Attacks in IoT-based Manufacturing Systems." *International Journal of Interactive Multimedia & Artificial Intelligence* 4.3 (2017).

[7] Backenstrass, T., et al. "Protection of ECC computations against side-channel attacks for lightweight implementations." *Verification and Security Workshop (IVSW), IEEE International*. IEEE, 2016.

[8] Bahnasawi, Mohamed A., et al. "ASIC-oriented comparative review of hardware security algorithms for internet of things applications." *Microelectronics (ICM), 2016 28th International Conference on*. IEEE, 2016.

ISSN: 2454-4248
04 – 08

[9] Gonzalez, Jesus David Terrazas, and Witold Kinsner. "Zero-Crossing Analysis of Lévy Walks and a DDoS Dataset for Real-Time Feature Extraction: Composite and Applied Signal Analysis for Strengthening the Internet-of-Things Against DDoS Attacks." *International Journal of Software Science and Computational Intelligence (IJSSCI)* 8.4 (2016): 1-28.

[10] Han, Guangjie, et al. "Security and privacy in Internet of things: methods, architectures, and solutions." *Security and Communication Networks* 9.15 (2016): 2641-2642.