The Security of Cloud Computing

Alexander Powell
College of William & Mary
Department of Computer Science
Email: ajpowell@email.wm.edu

Abstract—**This paper provides an overview of Elliptic Curve Cryptography. It also discusses its implementation over two finite fields. Additionally, we will discuss its benefits and shortcomings as well as the future potential of this type of public key cryptography.**

I. (20 POINTS) A SUMMARY OF THE RESEARCH FIELD YOU PLAN TO SURVEY.

Cloud computing, otherwise known as on-demand computing is a type of computing that is based on the internet and set up in such a way that it can share processors and data between computers and other devices on demand. One of the biggest benefits of cloud computing is that it allows companies to avoid infrastructure costs up front, and therefore give more attentions to other projects. Additionally, cloud computing offers the ability to get applications up and running faster, and makes it easier to manage projects and spend less time maintaining them. In recent years, cloud computing has been in high demand due to its power, performance, price points, scalability, and availability. The most significant developments in this field have come about in the past decade. In 2008, NASA released the first open-source software package for developing provate and hybrid clouds, called OpenNebula.

Currently, there are three major service models in the cloud infrastructure. Together, these models form a stack: infrastructure, platform, and software. The most basic cloud service model is called infrastructure as a service (IaaS). IaaS refers to online services that abstract the user from the details of infrastructure like physical computing resources, location, security, etc. To deploy their applications, cloud users install operating-system images oand their software on the cloud infrastructure. In this way, the user patches and maintains the operating system. In platform as a service (PaaS), vendors offer a development environment to application developers. The provider then develops toolkits and standards for development as well as channels for distribution. It is important to note that in this platform, the provider givers a platform, usually an operating system and development environment on which the developers can develop and run their software solutions on a cloud platform. Examples of PaaS include frameworks like Microsoft Azure, the Google App Engine, etc. There also exists specialized applications of PaaS. These include integration platform as a service (iPaaS) and data platform as a service (dPaaS). Furthermore, consumers of PaaS do not manage or control the cloud infrastructure of their network, only the deployed applications and configuration settings. The last service model is software as a service (SaaS). In this model, users gain access to application software and databases. Cloud providers manage the infrastructure and platforms that run the applications. SaaS gives businesses the potential to reduce IT operational costs by outsourcing hardware as well as software maintenance directly to the cloud provider.

II. (10 POINTS) JUSTIFICATION ABOUT WHY YOU PLAN TO SURVEY THIS FIELD.

Cloud computing is currently one of the fastest growing developments in the computing world. The economic case for it is extremely compelling, however there are a lot of concerns about the security challenges that it poses. Many businesses have accepted the economic case for cloud computing and more and more are moving towards similar solutions every day. Cloud computing providers can easily build large data warehouses for a relatively low cost to their customers. Also, as more individuals and businesses buy into these solutions, the resulting economies of scale increase revenue for cloud providers at a lower cost for cloud users. However, while cloud computing provides a convenient option for businesses to utilize, the security of the system has become it's greatest barrier to success. Currently, there are too many risks to service availability, data confidentiality, etc. for cloud computing to be presented as a feasible solution for most people, but the demand is nonetheless increasing as the price goes down. For this reason, it's important to evaluate the problems in cloud security and their implications. There are a number of concerns about the current state of cloud computing security. The biggest of these are access, compliance, location, recovery, viability, and availability. Access deals with who has administrator access to sensitive data as well as who makes decisions about hiring and supervision of such indivisuals. Compliance deals with potential issues like audits and security certifications from the outside. Unsurprisingly, the principle of location pertains to whether the cloud vendor allows for any control of the location of data. Recovery addresses the issue of whether data can easily be restored or recovered in the event of a disaster, and how much time would that take. Viability pertains to issues like what would happen if the client outlives the cloud vendor. Do they get their data back, and how is it returned to them. Finally, data availability means if their are periods when the vendor needs to change/update system requirements, is the data temporarily unavailable to the client.

Fortunately, the Internation Standard Organization (ISO), has published a number of themes that all security systems should be checked against. These themes are explained below.

- Identification & Authentication - Depending on what cloud model is being used, users must begin by providing necessary credentials so that they can be granted the permissions they need.
- Authorization - Authorization is used to ensure that integrity is maintained. Authorization is maintained by the system administrator to handle different priveleges and permissions of developers.
- Confidentiality - Confidentiality is essential in the world of cloud computing due to the accessible nature of public clouds. Making user profiles and data confidential ensures that information security protocols can be properly enforced at various stages down the line.
- Integrity - Integrity comes into play when users are accessing data. When employees are dealing with customer data there should always be some expectation of privacy.
- Non-repudiation - Non-repudiation means that the "author" of something won't be able to challenge the fact that they wrote something. This principle comes up a lot in public key cryptography, so that as long as everyone keeps their private key a secret, noone will be able to impersonate them. In this way, individuals are able to leave digital signatures, timestamps, and confirmation receipts on data they are using.
- Availability - Finally, availability is a key decision factor when choosing between private, public, or hybrid clouds. The service level agreement highlights the downside of availability in cloud resources.

III. (20 points) A classification scheme of the techniques in your selected field.

IV. (20 points) A classification of the techniques in your reference list with your classification scheme. You need to map each paper you cite to one of the classes.

V. (20 points) A summary of the trends in your selected field. Give evidences for your decision.

More work in the field of cloud computing has been done in the last decade than any previous amount of time. As we have already seen, cloud computing appears to be an attractive business solution to many. However, its greatest weakness is its security, which is a major concern depending on the customer. In providing a secure cloud computing solution, a big decision is to decide the type of cloud to be implemented. There are currently three main types of cloud deployment models: public, private, and hybrid clouds. These all offer different advantages and disadvantages in terms of performance and security.

- Public Clouds
  The model of the public cloud allows user access to the cloud through an interface like an internet browser. In general, it's either free of maintained on a pay-per-use system, meaning the client will pay more during periods of higher demand. In terms of what the user sees, there may appear to be no difference between public and private cloud deployment models. However, the implications for security can differ substantially depending on the service. Some of the more famous public cloud service providers like Amazon, Microsoft, and Google own and operate the entire infrastructure at their own private data warehouse. The data is access using an internet connection.
- Private Clouds
  The concept of a private cloud computing system is intended for a single organization, and is either managed internally or by some third party. The private cloud model raised security issues that must be addresses in order to prevent dangerous security loopholes. Furthermore, private cloud centers often require a lot of space and resources, in addition to a significant level of engagement by the business. Because of this, many individuals choose the public cloud route because they don't end up saving money or resources. In this way, private cloud models aren't as economical as it's public or hybrid cousins.
- Hybrid Clouds
  Unsurprisingly, a hybrid cloud is a mix between the public and private cloud models. Examples of scenarios when hybrid clouds may be beneficial include situations when private client data may be stored on a private cloud application, but then that application is connected to a business intelligence application deployed on a public cloud. Another example would be when a public cloud is used to meet temporary capacity needs that cannot be met by the private cloud. This technique is known as cloud-bursting, and it is employed when an application normally runs on a private cloud but automatically transfers (or bursts) to a public cloud when the demand for computing capacity reaches a certain level.

Talk about three major types of clouds. Maybe more on [X]aaS's More details about security risks.

VI. (10 points) A reference list with at least 20 references.

VII. Introduction

Elliptic Curve Cryptography (ECC) is a type of public key cryptography (PKC) which uses algorithms based on mathematical problems without an efficient solution. Some of the early PKC systems relied on factoring large integers composed of two or more large prime factors. Elliptic Curve Cryptography is a method of cryptography that relies on elliptic curves to provide security. The use of elliptic curves for purposes related to cryptography was proposed by both Neal Koblitz and Victor S Miller in 1985. By 2004, ECC algorithms had gained significant popularity.

VIII. Overview

In general, the first assumption followed when dealing with ECC is that the equation of an ellipse is written as

$$y2 = x^3 + ax + b \text{ where } 4a^3 + 27b^2 \neq 0$$

This equation is called the Weierstrass normal form for elliptic curves. Clearly, by changing the values for a and b the curve will take on different points. Also, in general elliptic curves are symmetric about the x-axis. Additionally, a point at infinity or ideal point will be part of the curve. This point is often denoted with a 0, giving the following definition for an elliptic curve

$$\{(x,y) \in \mathbb{R}^2 \mid y^2 = x^3 + ax + b,\ 4a^3 + 27b^2 \neq 0\} \cup \{0\}$$

The next step is to define a group over elliptic curves. To prove something is a group, it must satisfy the following properties: closure, associativity, identity, and inverse. In our case, a group over elliptic curves can be established because all points on the curve make up the elements of the group, the identity element is said to be the point at infinity or ideal point. The inverse of a point in the group is the one symmetric about the x-axis. Finally, addition is given by the rule $P + Q + R = 0$ where P, G, and R are three points on the same line. Just like a lot of other PKC systems rely on factoring large numbers, the security of ECC depends on the difficulty of what‘s called the Discrete Logarithm Problem.[1] In this problem, P and Q are two points on an elliptic curve such that $kP = Q$ where k is a scalar. Given P and Q, its incredibly difficult to solve for k if k is sufficiently large. Also, k is the discrete logarithm of Q to the base P. The discrete logarithm problem is believed to be a "hard" problem, meaning it has no known polynomial time algorithm. However, this belief has not been formally proved. This brings us to the main operation involved in the ECC system, point multiplication.

IX. Point Multiplication

The basic idea behind point multiplication in ECC is multiplying a point P on an elliptic curve with a scalar k to obtain another point Q on the same elliptic curve. There are two components to point multiplication: point addition and point doubling.[2] In point addition, two points J and K are added together to obtain a new point $L(J + K = L)$. In point doubling, a point is added to itself (or multiplied by 2) to obtain a new point $(2 \times J = J + J = L)$. An example of point multiplication with a point P and $k = 23$ is given as:

$$23P = 2(2(2(2P) + P) + P) + P.$$

Here, we can see that point multiplication utilizes point addition and doubling repeatedly to find the solution. This is referred to as the double and add method, although there are many other strategies. We can also view this algebraically. First, take two points on an elliptic curve, J and K, where $J = (x_J, y_J)$ and $K = (x_k, y_k)$. Now let $L = J + K$ where $L = (x_L, y_L)$, which gives us $x_L = s^2 - x_J - x_k$ and $y_L = -y_J + s(x_J - x_L)$ and $s = (y_J - y_k)/(x_J - x_k)$, where s is the slope of the line through points J and K. We can clearly see that is $K = -J$ meaning $K = (x_J, -y_J)$ then $J + K = 0$ where 0 is the point at infinity or ideal point.[1] Also, if K = J then it follows that $J + K = 2J$ and point doubling equations can be used. Additionally, we can see that the commutative property holds and $J + K = K + J$.

X. Finite Fields

In the previous examples, weve computed point multiplications for real numbers. These types of operations over the set of real numbers can be slow as well as inaccurate as a result of round off error. To make these operations faster and more efficient ECC is usually defined over two fields: a prime field F_P and a binary field F_2. On the prime field, the equation for the elliptical curve is slightly rewritten in its modular form:

$$y^2 \bmod p = x^3 + ax + b \bmod p, \text{ where } 4a^3 + 27b^2 \bmod p \neq 0.$$

In this case, the elements of the finite field are integers between 0 and $p - 1$. When ECC is operated over the prime field, the prime p is chosen in such a way that there is a large number of points on the elliptic curve to make the cryptography secure.[3] When generated over the prime field, the elliptic curve is not a smooth one, which causes difficulty in geometric reasoning about point multiplication. However, the algebraic rules still apply when using curves generated over F_P. When curves are generated over the binary field, the following equation is used:

$$y^2 + xy = x^3 + ax^2 + b, \text{ where } b \neq 0.$$

In this case, the elements of the field are integers with a length of at most m bits. Additionally, these numbers can be expressed as a binary polynomial (coefficients can only be 0 or 1) of degree $m - 1$. Like before, it is important to choose an m such that there is a sufficiently large enough number of points on the elliptic curve.[3]

XI. Key Sizes

One of the benefits to using ECC over other PKC systems is that ECC can use a smaller key size and still retain security.[4] Furthermore, this reduces storage and transmission requirements. For example, an ECC public key with 256 bits should provide an equivalent level of security to a 3072 bit RSA public key. Currently, the fastest algorithms to solve ECC codes require $O(\sqrt{n})$ iterations. This implies that the size of the field used should be twice the security parameter. For example, for a 128 bit code, an elliptic curve over the field F_q, where $q = 2^{256}$ should be used. To date, the most complex ECC system known to be broken has a 112 bit key for the prime field case and a 109 bit key for the binary field case.[2] Both of these cases took a matter of months to break.

XII. Public Key Algorithms

There are many encryption algorithms related to the field of ECC. Two of the most common are ECDH (Elliptic curve Diffie-Hellman) and ECDSA (Elliptic Curve Digital Signature Algorithm).

A. Elliptic Curve Diffie-Hellman

ECDH is a variation of the Diffie-Hellman algorithm that is used for elliptic curves. In reality, it's more of a key agreement protocol than an encryption algorithm, which means it tells us how keys should be generated and exchanges between users. The ECDH algorithm is used when there are two users who want to exchange information securely in such a way that a third user can intercept it but not decode it. To give a high level overview, there are three steps:

- To start, the two users attempting to exchange information securely, let‘s call them A and B both have individual public and private keys. However, these keys have been generated using the same base point on the same elliptic curve.
- Next, user A and B exchange their public keys through the third user, C, who they do not want decoding it. Note: it‘s assumed C can’t decode the keys because to do that would mean solving the discrete logarithm problem, which doesn’t have a polynomial time algorithm.
- Finally, A and B can both decode the information using their own private key and the other‘s public key.

It should be noted that the Diffie-Hellman is considered to be a “hard” problem. Furthermore, it‘s believed to be as hard as the discrete logarithm problem, although this has not been rigorously proven.[4] However, we know that it isn‘t any harder because solving the discrete logarithm problem would mean solving the Diffie-Hellman problem. ECDH is sometimes refered to as ECDHE, where the second E stands for Ephemeral meaning that the keys exchanged are only temporary, and not static. This is used when A and B need to generate their keys on the fly at the time the connection is established. Then the keys are later signed for authentication and exchanged.[4]

B. Elliptic Curve Digital Signature Algorithm

ECDSA attemtps to solve the problem where again there are two users A and B, and A wants to sign a message with their private key and B wants to validate the signature using A‘s public key. Also, only A should be able to produce valid signatures, but all parties should be able to check signatures. Here, A and B are using the same domain parameters. The ECDSA algorithm is a variation of the Digital Signature Algorithm applied to elliptic curves. ECDSA operates on the hashed message instead the actual message itself. There are an unlimited number of hash functions to choose from, but it is important that a cryptographically-secure one is chosen.

At a high level, the algorithm first generated a private key, which is then “hidden” using point multiplication. For B to verify the signature, A‘s public key, the hash code and the signature are needed. From this, two integers are calculated to form a new point on the elliptic curve.[4]

XIII. Comparison between ECC and RSA

We’ve seen the ECC does a good job with secure encryptions. However, one may ask why even use ECC at all if we already have good RSA encryption algorithms. One of the most significant benefits is that ECC requires a much smaller key size to guarantee the same level of security. In the table below, we see the corresponding key sizes (in bits) between the two strategies.

RSA key size	ECC key size
1024	160
2048	224
3072	256
7680	384
15360	521

From this table, there doesn‘t appear to be any sort of linear relationship between the RSA and ECC key sizes, which implies that ECC uses less memory and also that key generation and signing are considerably faster.[5]

In the equations for an elliptic curve given above, some coefficient values produce stronger curves than other, from a security perspective. Generally, a random seed is used in the domain parameters to generate the elliptic curve. The random seeds often come from digits of π, Euler’s number *e*, or the golden ratio.[4] These numbers are random because their digits are uniformly distributed. It is not common knowlegde of where the random seeds that the NSA and NIST come from. Many have speculated that these organizations have discovered a large class of weak elliptic curves and have tries all possibly seeds until they found a vulnerable curve. However, it‘s important to note that random and secure are not synonymous in the field, and there‘s nothing to be done if the algorithm is broken no matter how long the keys are. In this case, RSA is the stronger method because it does not require special domain parameters that can be tampered with. Therefore, RSA is still probably a better choice over ECC if we cant‘t trust authorities or can‘t construct the domain parameters ourselves.[6]

XIV. Conclusion

In conclusion, we have seen that there are many benefits to the ECC strategy of public key encryption. ECC provides an incredibly fast and efficient alternative to RSA methods due to it‘s efficiency in the use of memory. However, for an efficient implementation of ECC, it is important for the point multiplication algorithm and the field arithmetic to be efficient. Elliptic curves have many applications including not only data encryption but also digital signatures, pseudo-random generators, integer factorization algorithms, and many other tasks. Although a relatively recent invention (1985) but only gaining real prominence in the early 2000s, the applications to ECC are endless and the benefits over other traditional approaches are huge.

References

[1] Christof Parr and Cetink Koc, *Software Implementation of Elliptic Curve Cryptography over Binary Fields*, Springer Berlin Heidelberg, 2000

[2] Sheueling Chang Shantz, Hans Eberle, Arvinderpal Wander, and Arun Patel, *Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs*, Springer Berlin Heidelberg, Lecture Notes in Computer Science: 2004

[3] Anoop MS, *Elliptic Curve Cryptography: An Implementation Guide*, InfoSECWriters, 2011

[4] Andrea Corbellini, *Elliptic Curve Cryptography: A Gentle Introduction*, 2015

[5] Vivek Kapoor, Vivek Sonny Abraham, and Ramesh Singh, *Elliptic curve cryptography*, ACM Ubiquity 2008

[6] David J Malan, Matt Welsh, and Michael D. Smith, *A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography*, Sensor and Ad Hoc Communications and Networks, 2004