International Journal of Security (IJS)

ISSN : 1985-2312

VOLUME 4, ISSUE 4

PUBLICATION FREQUENCY: 6 ISSUES PER YEAR

International Journal of Security (IJS)

Volume 4, Issue 4, 2010

Edited By
Computer Science Journals
www.cscjournals.org

International Journal of Security (IJS)

Book: 2010 Volume 4, Issue 4
Publishing Date: 30-10-2010
Proceedings
ISSN (Online): 1985-2320

IJS Journal is a part of CSC Publishers
http://www.cscjournals.org

Published in Malaysia

Typesetting: Camera-ready by author, data conversation by CSC Publishing Services – CSC Journals, Malaysia

CSC Publishers

Editorial Preface

This is the fourth issue of volume four of The International Journal of Security (IJS). The Journal is published bi-monthly, with papers being peer reviewed to high international standards. The International Journal of Security is not limited to a specific aspect of Security Science but it is devoted to the publication of high quality papers on all division of computer security in general. IJS intends to disseminate knowledge in the various disciplines of the computer security field from theoretical, practical and analytical research to physical implications and theoretical or quantitative discussion intended for academic and industrial progress. In order to position IJS as one of the good journal on Security Science, a group of highly valuable scholars are serving on the editorial board. The International Editorial Board ensures that significant developments in computer security from around the world are reflected in the Journal. Some important topics covers by journal are Access control and audit, Anonymity and pseudonym, Computer forensics, Denial of service, Network forensics etc.

The coverage of the journal includes all new theoretical and experimental findings in the fields of computer security which enhance the knowledge of scientist, industrials, researchers and all those persons who are coupled with computer security field. IJS objective is to publish articles that are not only technically proficient but also contains information and ideas of fresh interest for International readership. IJS aims to handle submissions courteously and promptly. IJS objectives are to promote and extend the use of all methods in the principal disciplines of computer security.

IJS editors understand that how much it is important for authors and researchers to have their work published with a minimum delay after submission of their papers. They also strongly believe that the direct communication between the editors and authors are important for the welfare, quality and wellbeing of the Journal and its readers. Therefore, all activities from paper submission to paper publication are controlled through electronic systems that include electronic submission, editorial panel and review system that ensures rapid decision with least delays in the publication processes.

To build its international reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJS. We would like to remind you that the success of our journal depends directly on the number of quality articles submitted for review. Accordingly, we would like to request your participation by submitting quality manuscripts for review and encouraging your colleagues to submit quality manuscripts for review. One of the great benefits we can provide to our prospective authors is the mentoring nature of our review process. IJS provides authors with high quality, helpful reviews that are shaped to assist authors in improving their manuscripts.

Editorial Board Members
International Journal of Security (IJS)

Editorial Board

Table of Content

Volume 4, Issue 4, October 2010

Pages

A Performance Analysis of Chasing Intruders by Implementing Mobile Agents

Omid Mahdi Ebadati E omidit@gmail.com
Dept. of Computer Science
Hamdard University
New Delhi, India

Harleen Kaur harleen@jamiahamdard.ac.in
Dept. of Computer Science
Hamdard University
New Delhi, India

M. Afshar Alam aalam@jamiahamdard.ac.in
Dept. of Computer Science
Hamdard University
New Delhi, India

Abstract

An Intrusion Detection System in network fetches the intrusions information from systems by using Mobile Agents aid. Intrusion Detection System detects intrusions based on the collected information and routes the intrusion.
The intelligent decisions on communications permit agents to gain their goals more efficiently and provide more survivability and security of an agent system. The proposed model showed a formal representation of information assurance in agent messaging over a dynamic network by probability of redundant routes.
The proposed Intrusion Detection System, chase intruders and collect information by Mobile Agents. Our proposed architecture is an information exchange method and chasing intrusion along with a method by implementing Mobile Agents.

Keywords: Intrusion Detection System (IDS), Hybrid Intrusion Detection System (HyIDS), Mobile Agent (MA).

1. INTRODUCTION

Computer network attacks generally are divided into two types: break in from outside of local area network and the other attack from inside the local area network. Therefore, it is rarely happen in either case that intruders can directly attacks from their own host [1]. The reason is obvious: intruders want to conceal their origin.
Intruders prefer to attack the minimum secured hosts first, then smoothly approaching secured host with stronger protection, consequently try up to reach their target hosts. Typically the users or administrators do not notice about intrusion attacks, either on the target hosts or the intermediate hosts, and the administrators cannot chase the origin of an intrusion even the intrusion has been detected or the network connection has logged out. Intrusion Detection has

the ability to analyze data in real time to detect the intruder and block the attacks when they occurred.
There are various types of Intrusion Detection Systems and each one has different activity and different ways of detection. Practically detecting the intrusion is more complex than a simple definition.
With the Mobile Agents over a network environment that support it, it can be penetrated possibly harmful agents that called intruders. By increasing the damages of intruder's, motivation of a large amount of research on detection part especially by Mobile Agent team, also increased. The Mobile Agent team, deploy their chasing over the network just by detection of an intruder.
Here, we are defining the different types of Intrusion Detection System:
A Network Intrusion Detection System (NIDS) which identify the intrusion by monitoring the network traffic; it has an independent platform of detection [2] and can monitor multiple hosts. Network Intrusion Detection Systems approach access to network traffic by connecting to a hub network or switch, to configure the mirroring port, or network tap such as a snort, which is an example of NIDS.
A Protocol based Intrusion Detection System (PIDS) is based on communication protocol between different connected devices, such as user or system and the server. PIDS has agents that settle down on server front-end that analyzes and monitor the network traffic.
Other type of IDS is Application Protocol based Intrusion Detection System (APIDS). The APDIS is based on a system or agents that located within a number of servers, and try to analyze and monitor the communication on specific application protocols.
A Host based Intrusion Detection System (HIDS) is basis on an agents on a host [12,13,14] that identify intrusions by various measurement like file system modifications such as binaries, password files, analyzing system call, application logs and other host activities and states.
The last model of Intrusion Detection System is Hybrid Intrusion Detection system (HyIDS). HyIDS is a combination of two or more approaches of Intrusion Detection Systems. In case of implementing a combine specification of intrusion detections, we can have a new comprehensive view of network detection.

2. RELATED WORK

Using Mobile Agents in Intrusion Detection System being conducted by various researchers including Wayne Jansen, Peter Mell, which worked on reification of their own model on specific network [15], Mohamad Eid, proposed immunity components [16], Omid Mahdi Ebadati, Harleen Kaur, M. Afshar Alam, proposed secured route in NIDS [17], Christopher Krugel, Thomas Toth conducted their work on Sparta and Micael [18], Shunji Okazawa, Midori Asaka, Atsushi Taguchi, proposed detection in an intermediate node [19], Wayne A. Jansen, proposed a new design with more robust [24].
By introducing the intrusion detection to the industry, it is become a part of detection technology, and different companies commercially perfecting the existing intrusion detection techniques. Finally the research part focuses on the most unsealed part that those are:
1- Mechanisms of attack response.
2- Architecture for distributed Intrusion Detection Systems and standards of intrusion detection inter operability.
3- New paradigms for performing intrusion detection by Intrusion Detection System.

3. INTRUSION DETECTION SYSTEM ARCHITECTURE

In a large scale to deploying an Intrusion Detection System over a network, network traffic will be ultimately high though the high volume of the system logs will regularly transferred, in this case large number of information which has been collected, are unrelated to intrusion. Therefore in such cases Intrusion Detection System on a large network environment does not have efficient functionality. For the solution of this problem we adopted a Mobile Agent paradigm in developing IDS. Mobile Agents autonomously [10] route to the target systems for just collecting the intrusion related information and eliminate the requirement of transferring logs to the server.

Through the TCP/IP protocol Intrusion Detection System can deploy on the local area network. The proposed Intrusion Detection System typically consists of various elements like a sensors, administrator, notice boards, flag/message boards, information collector agents, and chasing agents.

Information collector agents— Information collector agents during the normal process gathering the information from the specific network. This information which has been collected may contain intrusion also but the recognition of this matter is unrelated to this part.

Administrator— The administrator of intrusion detection has the duty of recognition of intrusions. After information collector agents gathered the information over the specify network, the administrator try to detect the intrusions among them. The administrator manages the notice boards and mobile agents and tries to making interface between system and administrators. The detection part according to the information weight which entered by agents on the notice board and exist in manager accumulator can find out the intrusion. If the weights are more than a set threshold administrator concludes that it is an intrusion.

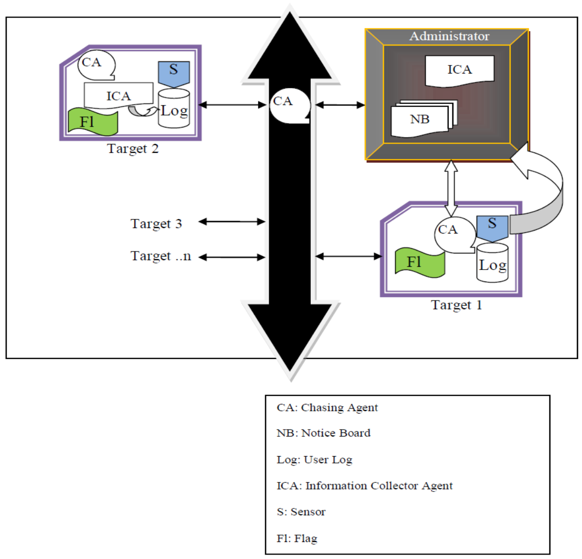


FIGURE 1: Intrusion Detection System Architecture

Notice boards and sensors— Notice boards by the aim of sensors try to specify the information and get direction to the administrator.

The sensor on each agent by logging which conduct by system monitoring send the report of collected logs to the administrator.

Chasing agent— Chasing agent is specifically to route and trace the path of intrusion and try to locate the origin of intrusion. This chase can be done by detection of remote logged on, to the target host. The compromised nodes in network can detect by chasing agent.

Intrusion Detection System functionality— The speed of detection part which depend on platform of detection and different algorithm that can use in the system shows the performance of IDS. If the functionality of IDS wants to be acceptable, speed of detection and reliability of that must be increase. The requirement performance of IDS can depend on, real time detection of anomalous event and security breaches and an immediately report is necessary to reduce the corruption and loss on the network to have the confidential data.

The Intrusion Detection System should not busy by different deployment on the system and not to interfere with the normal operation on it. The important thing is that, the agents aware of the consumption of network resources, and the tradeoff between additional levels of security that monitoring the network and checking the agents performance.

Scalability of Intrusion Detection System due to the new computing devices which added to the network is another issue of performance, and IDS must be able to handle the communication load and these computations [8].

4. FUNCTIONALITY OF IDS IN MOBILE AGENTS

There is a certain platform of Mobile Agent that can work on variety of hosts and systems. In this platform not all but at least hosts and network devices must be installed with Mobile agent platform. Mobile Agent platform give the capability of implementation various applications and simply can assume companies and organizations [9]. Mobile Agents are not preinstalled on all the systems because, it has not that much popular to be, and so if it is, it needs to contrast for IDS scheme and assume on host based Intrusion Detection System. Generally this scheme is too expensive and it is unusual to install Mobile Agent platform or Java virtual machine on all hosts for a general purpose.

Mobile Agent Merits

Previously Mobile Agents have not the ability to move in different environments. As virtual machines and interpreter has been introduced, it got the possibility for heterogeneous environment and platform but however there is a limitation support for preservation and resumption of the execution state.

Mobile Agents although carry a deal of autonomy and role well in operating disconnection and the failure of platforms or home platform, that the agent confidentially provide the security services and reduce their functionality.

The fault tolerant of Mobile Agent can be increased when agents moving from a machine to another machine therefore, the Mobile Agents always working on the trusted machine and restrict on its functionality, so its environments will be a trusted platform and a safe home.

Mobile Agents platform designers mostly face tradeoff between fault tolerance and security. An example of security risk can be multi hop in Mobile Agent that cause of different types of architecture which have been built in client server centralization models that need those agents return to a central server before moving to another machine/host machine [20]. Even though having a security risk of Mobile Agents shows the vulnerable and failing of MA for central server.

There is some keynotes about Mobile Agents like network latency over coming, network load reduction, execution and autonomy asynchronization, composition and structure, scalability, dynamic adoption, fault tolerance behavior and robust, hydrogenise environment operations.

Mobile Agents have many capabilities to enhance with Intrusion Detection System technology. The first obvious one is mobility which is the most important characteristics of MA. Agent lending capability to Intrusion Detection System can be another one. Agent application and technology mimic collections of intelligent individual and autonomous. Individual classes have special applications and each class can operate individually and independently than each other and each one individually exchange the information and talk to other when they meet individually. This paradigm is clearing the traditional programming where Master Logic Unit controls slave unit that have not autonomy and work according to command of Master Logic Unit.

Traditional approach which work on multiple units with set communication channel and set duties cannot response well, suppose each unit rely on other unit to perform their job and there is not central controller. In case of a unit seas the other unit are not enough intelligent to function and sort out the problem. The traditional distribution programming paradigm upon function reliability to component can only work well, and even by using redundant components a small number of back offs can attack and make it disable with attackers. Even though the implementation of the traditional design and solution can efficiency solves many problems. A great contrast to traditional design is agent technology that attempt to give understanding of environment along with independently decision making and authority to each agent.

Chasing Agents on Network Environment

The agents take their way and decide their destination and their path to them by use of notice board. Chasing agents perform as follow:

First of all, agents know that on each system there is a process and user ID, therefore dispatching process to a target system by chasing a process ID [11] is the first perform.

Secondly, the chasing agent tries to find out the logs on target systems, if it detects any, it will determine its destinations from the information on user’s log.

Thirdly, referring to the notice board on the target system is the next refer of chasing agent.

Fourthly, it will check whether on notice board there is any information related to login session that intend the agent to chase, if there is any related to that session on the notice board and its already exist, it shows that another agent has already tried and traced this user, so the chasing agent update the notice board, and returns to the administrator.
Fifthly, if there is no information pertains to login session, the agent enters to such information and tries to get into the target system from the user logged. If the agent comprehensive that the origin of intrusion is that the target system which has been chased, the chasing agent return to the administrator.
Sixthly, if not, the steps one to six again should repeat by the agent.

Notice board information

The information can enter by chasing agent on the notice board of target system are: name of the system, process ID, the user logs timing, leaving process ID or the name of target and where it has been chased by agent, chasing agent ID, chasing agent trace time.
In case, if the chasing agent wants to discontinue a trace, on the notice board it should be enter its ID and time of performing. It will help when the agent return to the administrator, because the chasing route and the reason has been logged already.
For above case (agent's return) we can have three possibilities and they are: if the chasing is not perform so the intrusion cannot chase, mostly is the cause of not installation the Intrusion Detection System on the following system, the other reason can be whether the system itself is the origin of intrusion and the last can be chasing that intrusion by another agent.
As it has mention above also, if another agent is chasing the intrusion the concurrent agent send the name and ID of that agent of the following system to the administrator. The trace part is shown in Figure 2.

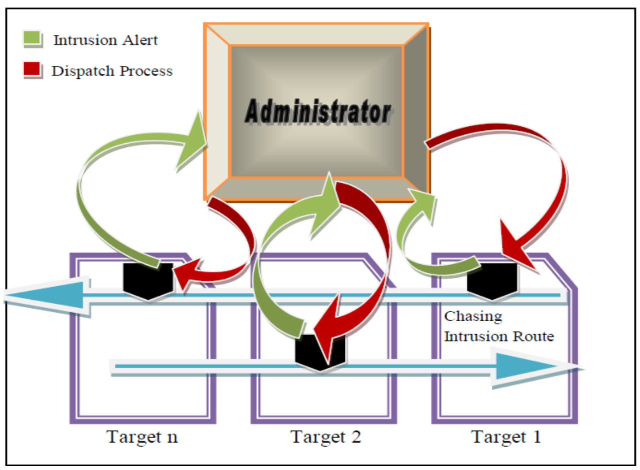


FIGURE 2: Intrusions Chasing

5. EXPERIMENT AND EVALUATION

Optimal Time Algorithm for Contiguous Search in Trees

This section is devoted to the description of a linear-time algorithm returning an optimal contiguous search strategy for weighted trees.
Let $T = (V, E)$ be a tree with n nodes. The quest for a minimal strategy can be restricted to monotone strategies satisfying the "single entry point" constraint, i.e., initially, at time $t = 0$, all searchers are in the same node x_0 called home base, and the first step consists to clear a link incident to the home base. The choice of the home base a_ects is the number of searcher of monotone strategies. Hence, let $cs_x(T)$, called search number of T from node x, denote the minimum number of searchers of monotone strategies starting from $x \in V$.
$cs(T) = min_{x \in V} \{cs_x(T)\}$ [23, 21, 22]. Any monotone strategy will induce an ordering on the neighbors (and thus on the incident links) of each node, where the ordering depends on whether one neighbor becomes occupied (and thus safe) before another. This ordering depends solely on

the choice of the starting node x, and corresponds to the parent/child relationship in T_x where T_x denotes the tree T when rooted in node x. Hence, in the following, we will refer to a strategy for T starting from x also as the strategy for T_x, and alternatively denote $cs_x(T)$ by $cs(T_x)$.

Given a rooted tree T_x and one of its nodes y, let $T_x[y]$ denote the subtree of T_x rooted in y, and let $cs(T_x[y])$ denote the contiguous search number of $T_x[y]$ from y. The most important property of monotone strategies is given by the following lemma.

Let $y_1, y_2; : : : ; y_k$ be the $k \geq 2$ children of y in T_x, and assume, w.l.o.g., that $cs(T_x[y_i]) \geq cs(T[y_{i+1}])$ for all $i < k$. Then $cs(T_x[y]) = max\{cs(T_x[y_1]; cs(T_x[y_2]) + \omega(y)\}$:

Proof. Clearly $cs(T_x[y]) \geq cs(T_x[y_1])$, otherwise $T_x[y_1]$ cannot be cleared. If $cs(T_x[y_1]) > cs(T_x[y_2]) + \omega(y)$, then $cs(T_x[y_1])$ searchers suffice to clear $T_x[y]$ by visiting y_1 last among the children of x, and by letting $\omega(y)$ searchers occupying node y while the other subtrees are visited. Indeed, from Eq. 1, $\omega(\{y,y_i\}) \leq \omega(y_i) \leq cs(T_x[y_i])$ for every i.

Hence, let $cs(T_x[y_1]) < cs(T_x[y_2]) + \omega(y)$. Let β be a contiguous search strategy which uses $b = cs(T_x[y_2]) + \omega(y) - 1$ searchers to clear $T_x[y]$. If $Tx[y_2]$ is cleared before $T_x[y_1]$, while the $cs(T_x[y_2])$ searchers are clearing $T_x[y_2]$, y will be occupied by at most $\omega(y) - 1$ searchers, and incident to $\{y,y_1\}$ which is contaminated. Thus, β does not satisfy the condition of Theorem 1. Similarly, if $T_x[y_1]$ is cleared before $T_x[y_2]$, then, while the $cs(T_x[y_1])$ searchers are clearing $T_x[y_1]$, at most $\omega(y)-1$ searchers will be at y since by definition $cs(T_x[y_1]) \geq cs(T_x[y_2])$, and $b = cs(T_x[y_2])+ \omega(y)-1$.

Since y is incident to $\{y,y_2\}$ which is contaminated, y becomes unsafe, and β is hence not monotone. Therefore, strictly more than $cs(T_x[y_2])+ \omega(y)-1$ searchers are required for a monotone strategy if $cs(T_x[y_1]) < cs(T_x[y_2])+ \omega(y)$. On the other hand, $cs(T_x[y_2])+ \omega(y)$ searchers clearly suffice to clear $T_x[y]$ by visiting y_1 last among the children of y, since at least $\omega(y)$ will stay at y making it safe while the other subtrees are visited. Links incident to y are cleared since, again from Eq. 1, $\omega(\{y,y_i\}) \leq \omega(y_i) \leq cs(T_x[y_i])$ for every i.

A straightforward application of Lemma allows to compute $cs(T_x)$ in $O(n)$ time, resulting in an $O(n^2)$ time algorithm for computing $cs(T)$ for any tree T. We show that the complexity can be reduced to $O(n)$, and more importantly that a minimal search strategy can also be computed in linear time.

Search /* Returns a strategy α */

(a) Set $\alpha = \varnothing$; /* empty sequence */

(b) Choose a node x such that $cs(T_x)$ is minimum;

/* start with $cs(T)$ searchers at the homebase*/

Let $y_1; y_2; : : : ; y_{deg}(x)$ be the $deg(x)$ neighbors of x where, w.l.o.g., $cs(T_x[y_i]) \geq cs(T_x[y_i+1])$;

(c) For $i = deg(x)$ down to 1, apply

Move$(x, y_i, cs(T_x[y_i]))$;

Move (u, v, q) /* searching a subtree rooted at v */

(1) $\alpha = \alpha|(u, v, q)$ where | denotes the concatenation operation; /* transfer q searchers from u to v */

Let $w_1; : : : ;w_{deg(v)-1}$ be the $deg(v) - 1$ children of v in T_x where, w.l.o.g., $cs(T_x[w_i]) \geq cs(T_x[w_{i+1}])$;

(2) For $i = deg(v) - 1$ down to 1, apply

Move$(v, w_i, cs(T_x[w_i]))$;

(3) $\alpha = \alpha \mid (v, u, q)$; /* return q searchers from v to u */

Attack Detection

The goal of Intrusion Detection System is detecting intrusion as many as possible because always detecting all intrusions is not possible, and IDS tries to precise the route of chasing of intrusion, to make it efficiently.

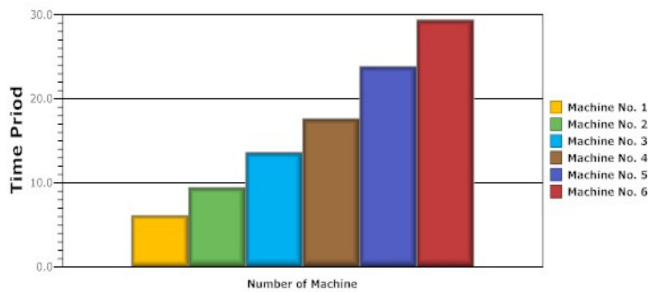
By increasing the internet accessibility, intrusions and cracking tools that distributed on the internet the appropriate evaluation of IDS is much difficult to examine. Another important factor that classifies the internet connection also is bandwidth of connected internet network. In proposed Intrusion Detection System we try to obtain such cracking tools that aimed at local attacks on the net and because of our limitation in trend of cracking tools which are available, we just try with Linux Fedora operating system machines with the limit number of network speed (less than six hundred Kbps). Our evaluation results has been tested the buffer overflow %54, password sniffing %7, route shell execution %3, file mode changing %2, file creation %27.4, routing %2.5 and administrator password sniffing %4.

Mobile Agents Performance Analysis

To initial a tracing and established a performance of an agent the Intrusion Detection System should determine the number of trigger. Generally the number of chasing agent and number of occur trigger on the administrator are same. In our investigation rate the whole events per day it was around (39 in 105617), so the rate of 0.000369 events per day is investigated on our machines. The average size of chasing agent excluding the information is 1.9 KB and 2.3 KB including the information for information collector agents and 4.1 KB for chasing agent.

The time period since a sensor trigger till chasing agent pertaining that return to administrator contained in an intrusion route in each case, there is a number of target system which is measured. Time period for agents' authentication also is calculated and it shows that including encryption and authentication, it takes around 1.4 times longer than when we have only authentication. The measured time is contain the process of chasing agent on each target and the transportation time period between targets and agent which become too short by using proposed algorithm, and it is around 0.07 second.

The chasing agents round trip, time period on a number of machines is as follows (Figure 3).



	Number of Machine
Machine No. 1	6.114
Machine No. 2	9.452
Machine No. 3	13.601
Machine No. 4	17.625
Machine No. 5	23.845
Machine No. 6	29.364

FIGURE 3: Chasing Agent, Round Trip-Time Period Diagram

6. CONCLUSIONS

Mobile Agent technology offers much related field in Intrusion Detection System. The autonomous components and MA seem an obvious usefulness in IDS and other applications as well.

However practically, hardly we can find out the technology of Mobile Agent and its beneficiary, but the technology is provided valuable capabilities and although in practice moves a running program from a platform to another one, without Mobile Agent system face us the barriers.

Mobile Agents can enter main stream by shows its performance, security, emerging technology, and widely use in different hardware platforms.

In this paper we described how intrusions chased by emerging Mobile Agents and Intrusion Detection System. However Intrusion Detection System does not indiscriminately in entire the network collect the information, but collect pertain information by chasing the logged in user.

The proposed architecture of Intrusion Detection System by implementing chasing route optimal time algorithm is briefly explained in this paper. The round trip time on a number of machines for chasing the agent shows that the round trip type is not that, much increased when the numbers of machines are increased, and the ramp of increasing is not as same as number of machine's ramp.

Proposed Intrusion Detection System does not collect unrelated information to intrusions and just focused on the amount of information which is gathered by agents, and administrator which has been approved them, consequently it will help to reduce the using of the system.

7. REFERENCES

[1] W. R. Cheswick, S. M. Bellovin. "Firewalls and Internet Security: Repelling the Wily Hacker," Addison Wesley Publishing Company (1994)
[2] Northcutt, S. Intrusion Detection - The Big Picture. SANS Institute, 1999, www.sansstore.org.
[3] Whitman, Michael, Herbert Mattord. Principles of Information Secuirty. Thomson, Canada, pp. 290-301 (2009)
[4] Anderson, Ross. Security Engineering. Wiley, New York, pp. 387-388 (2001)
[5] Kohlenberg, Toby (Ed.), Alder, Raven, Carter, Dr. Everett F. (Skip), Jr., Foster, James C., Jonkman Marty, Raffael, and Poor, Mike, "Snort IDS and IPS Toolkit," Syngress (2007)
[6] Barbara, Daniel, Couto, Julia, Jajodia, Sushil, Popyack, Leonard, and Wu, Ningning, "ADAM: Detecting Intrusions by Data Mining," Proceedings of the IEEE Workshop on Information Assurance and Security, West Point, NY (2001)
[7] M. Asaka, S. Okazawa, A. Taguchi and S. Goto. A method of tracing intruders by use of mobile agents. In 9th Annual Conference of the Internet Society (INET'99), 1999
[8] G. G. Helmer, J. S. K. Wong, V. Honavar and L. Miller. Intelligent agents for intrusion detection. In IEEE Information Technology Conference, 1998
[9] W. Jansen, P. Mell, T. Karygiannis and D. Marks. Mobile agents in intrusion detection and response. In 12th Annual Canadian Information Technology Security Symposium, 2000
[10] R.Gray, David Kotz, George Cybenko and Daniela Rus, "Security in a multiple-language mobile-agent system," In Giovanni Vigna (Ed.), Lecture Notes in Computer Science: Mobile Agents and Security, 1998
[11] W. R. Cheswick, S. M. Bellovin. "Firewalls and Internet Security: Repelling the Wily Hacker”. Addison Wesley Publishing Company (1994)
[12] Computer Operations, Audit, and Security Technology (COAST). "Introduction to Intrusion Detection”, www.cs.purdue.edu/coast/intrusion-detection/introduction.html
[13] Computer Security Institute. "The Cost of Computer Crime”, www.gocsi.com/losses.htm.
[14] Computer Security Institute. “Intrusion Detection Resources”, www.gocsi.com/intrusion.htm.
[15] Wayne Jansen, Peter Mell, “Applying Mobile Agents to Intrusion Detection and Response”. NIST Interim Report, 1999
[16] M. Eid. “A new Mobile Agent-based Intrusion Detection System Using Distributed Sensors”. In Third FEA Student conference, American University of Beirut, 2005
[17] Omid Mahdi Ebadati E., Harleen Kaur and M. Afshar Alam.“A Secure Confidence Routing Mechanism Using Network-based Intrusion Detection Systems”. In Second International Conference on Wireless Information Networks & Business information System (WINBIS’10), Nepal, 2010
[18] Christopher Krugel, Thomas Toth. “Applying Mobile Agent Technology to Intrusion Detection”. In ICSE Workshop on Software Engineering and Mobility”, 2001
[19] Midori Asaka, Shunji Okazawa and Atsushi Taguchi. “A Method of Tracing Intruders by Use of Mobile Agents”, Wasedo University, 1999
[20] “Jumping Beans Security,” Ad Astra Engineering, www.jumpingbeans.com/Security.htm
[21] Lali Barri`ere, Paola Flocchini, Pierre Fraigniaud, and Nicola Santoro. “Capture of an Intruder by Mobile Agents”. In SPAA’02, Winnipeg, Manitoba, Canada, 2002
[22] D. Bienstock, P. Seymour. Monotonicity in graph searching. Journal of Algorithms, 12, 239-245, 1991
[23] J. Ellis, H. Sudborough, J. Turner. The vertex separation and search number of a graph. Information and Computation, 113(1):50-79, 1994
[24] Wayne A. Jansen. “Intrusion Detection with Mobile Agents”. Mobile Agents Systems, National Institute of Standards and Technology, 2002, www.nist.gov

Preference of Efficient Architectures for GF(p) Elliptic Curve Crypto Operations using Multiple Parallel Multipliers

Adnan Abdul-Aziz Gutub aagutub@uqu.edu.sa

Center of Excellence in Hajj and Omrah Research,
Umm Al-Qura University P.O. Box: 6287, Makkah 21955, Saudi Arabia

Abstract

This paper explores architecture possibilities to utilize more than one multiplier to speedup the computation of GF(p) elliptic curve crypto systems. The architectures considers projective coordinates to reduce the GF(p) inversion complexity through additional multiplication operations. The study compares the standard projective coordinates (X/Z,Y/Z) with the Jacobian coordinates (X/Z^2,Y/Z^3) exploiting their multiplication operations parallelism. We assume using 2, 3, 4, and 5 parallel multipliers and accordingly choose the appropriate projective coordinate efficiently. The study proved that the Jacobian coordinates (X/Z^2,Y/Z^3) is preferred when single or two multipliers are used. Whenever 3 or 4 multipliers are available, the standard projective coordinates (X/Z,Y/Z) are favored. We found that designs with 5 multipliers have no benefit over the 4 multipliers because of the data dependency. These architectures study are particularly attractive for elliptic curve cryptosystems when hardware area optimization is the key concern.

Keywords: Modulo multipliers, Elliptic curve cryptography, Jacobian projective coordinates, Parallel multipliers crypto hardware.

1. INTRODUCTION

Elliptic Curve Cryptosystem (ECC) is a security system based on the discrete logarithm problem over points on an elliptic curve, proposed in 1985 by Victor Miller [1] and Niel Koblitz [2]. Although nowadays, ECC just exceeded\ 20 years old, its reliability is still suspect, with no significant breakthrough in determining weaknesses in the algorithm [3, 4]. In fact, the ECC problem appears very difficult to crack, implying that key sizes can be reduced in size considerably, even exponentially [5], particularly when compared to the key size used by other popular cryptosystems. This makes ECC become a promising practical replacement to the RSA, one of the most accepted public key methods known [6]. ECC promises to offer the same level of security as RSA but with much smaller key size. This advantage of ECC is being recognized recently where it is being incorporated in many standards [4, 28, 31]. In 1999, the Elliptic Curve Digital Signature Algorithm was adopted by ANSI, and it is now included in the ISO/IEC 15946 draft standards. Other standards that include Elliptic Curves as part of their specifications are the IEEE P1363 [7], the ATM Forum [8], and the Internet Engineering Task Force [9].

ECC systems can be implemented in software as well as hardware [10-20]. Hardware is preferred due to its better speed and security [5, 14, 15, 30]. Software, on the other hand, provides flexibility in the choice of the key size [13], which is also a feature adopted in hardware using “scalable multipliers” as clarified in [26, 29]. For cryptographic applications, it is more secure to handle the computations in hardware instead of software. Software-based systems can be interrupted and trespassed by intruders more easily than hardware, jeopardizing the whole application security [21].

Several ECC hardware processors have been proposed in the literature recently for Galois Fields GF(p) including GF(2^k) [11, 12, 15, 18-20, 26, 28-31]. The design of these processors is based on representing the elliptic curve points as projective coordinate points [11, 3, 15, 18, 26] in order to eliminate division, hence inversion, operations. It is known that adding two points over an elliptic curve requires a division operation, which is the most expensive operation over GF(p) [3, 22]. There are several candidates for projective coordinate systems. The choice thus far has been based on selecting the system that has the least number of multiplication steps, since multiplication over GF(p) is a common operation and the next most time consuming process in ECC.

In this paper we propose that the choice of the projective coordinate system should also depend on its inherent parallelism. High-speed crypto processors are crucial for today's security applications [21]. It will be proven in our work that parallelism can be a practical solution for meeting this requirement. We recommend using scalable GF(p) multipliers reported in [23] since they lead to wide range of hardware flexibility and trade-offs between area and time, compared to conventional GF(p) multipliers. The scalable multipliers allow the VLSI designer to choose between area and time as required by the application. Scalable multipliers are implemented in digit serial fashion, which is more efficient than both unpipelined and pipelined parallel multipliers for algorithms with repeated multiplications such as that found in ECC. It is worth noting that using pipelined parallel multipliers is not efficient for ECC where the multiplication of any iteration cannot begin before the multiplication operation of the previous iteration is completed. Also note that any ECC processor must implement the procedures of projective coordinates efficiently since they are the core steps of the point operation algorithm.

The main contribution of this paper can be viewed at the architectural level to make it utilize the parallelism within the projective coordinate procedure efficiently. The outline of the paper is as follows. In Section 2, we provide a brief theoretical background to elliptic curve cryptography, followed by an illustration of encryption and decryption. Section 2 also, outlines the algorithm used for ECC multiplication which is the basic concept behind using elliptic curve in cryptography. The elliptic curve point addition and doubling are elaborated using projective coordinates in Section 3, followed by the description of the proposed possible parallelization toward hardware architectures in Section 4, which will present the modeling and scheduling of data flow studies. The architecture efficient controller choice and area time cost of the different hardware is presented in Section 5. This is followed by the conclusions of the paper in Section 6.

2. ELLIPTIC CURVES OVER GF(P)

2.1 Theoretical Background

It will be assumed that the reader is familiar with the arithmetic over elliptic curves. For a good review the reader is referred to [3]. The elliptic curve arithmetic of GF(p) is the usual *mod p* arithmetic. The elliptic curve equation over GF(p) is:

$$y^2 = x^3 + ax + b \text{ ; where } p > 3,\ 4a^3 + 27b^2 \neq 0, \text{ and } x, y, a, b \in GF(p).$$

There is also a single element named the point at infinity or the zero point denoted 'φ'. By adding this point, the projective version of the curve is obtained. If P and Q are two points on the elliptic curve, a third point which is the intersection of the curve with the line through P and Q can be uniquely described. If the line is tangent to the curve at a point, then that point is counted twice; and if the line is parallel to the y-axis, we define the third point as the point φ (zero point). Exactly one of these conditions holds for any pair of points on an elliptic curve. If a point on the elliptic curve is to be added to another point on the curve or to itself, some special addition rules are applied, depending on the finite field used.

The addition rules in this field GF(p) are as follows:

$\varphi = -\varphi$

$(x, y) + \varphi = (x, y)$

$(x, y) + (x, -y) = \varphi$

The addition of two different points on the elliptic curve is computed as shown below:

$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$; where $x_1 \neq x_2$

$\lambda = (y_2 - y_1)/(x_2 - x_1)$

$x_3 = \lambda^2 - x_1 - x_2$

$y_3 = \lambda(x_1 - x_3) - y_1$

The addition of a point to itself (doubling a point) on the elliptic curve is computed as shown below:
$(x_1 , y_1) + (x_1 , y_1) = (x_3 , y_3)$; where $x_1 \neq 0$
$\lambda = (3(x_1)^2 + a) /(2y_1)$
$x_3 = \lambda^2 - 2x_1$
$y_3 = \lambda(x_1 - x_3) - y_1$

We assume that the squaring calculation has the same complexity as multiplication. To add two different points in GF(p) we need: six additions, one inversion, and three multiplication operations. To double a point we require: four additions, one inversion, and four multiplication computations. The GF(p) point operations will be discussed for ECC crypto processors in section 5.

2.2. Encryption and Decryption

There are many ways to apply elliptic curves for encryption/decryption purposes [3]. In its most basic form, users randomly select a base point (x,y), lying on the elliptic curve E. The plain text (the original message to be encrypted) is coded into an elliptic curve point (x_m,y_m). Each user selects a private key 'n' and computes his public key P = n(x,y). For example, user A's private key is n_A and his public key is $P_A = n_A(x,y)$.

For anyone to encrypt and send the message point (x_m,y_m) to user A, sender needs to choose a random integer R and generate the ciphertext: $C_m = \{R(x, y) , (x_m, y_m)+ kP_A \}$.
The ciphertext pair of points uses A's public key, where only user A can decrypt the plain text using his private key. To decrypt the ciphertext Cm, the first point in the pair of C_m, R(x,y), is multiplied by A's private key to get the point: n_A (R(x,y)). Then this point is subtracted from the second point of C_m, the result will be the plain text point (x_m,y_m). The complete decryption operations are:

$$((x_m,y_m)+RP_A) - n_A(R(x,y))=(x_m,y_m)+R(n_A(x,y))-n_A(R(x,y))=(x_m,y_m)$$

The most time consuming operation in the encryption and decryption procedure is finding the multiples of the base point, (x,y). The algorithm used to implement this is discussed in the next section.

2.3. Point Operation Algorithm

The ECC algorithm used for calculating nP from P is based on the binary representation of n, since it is known to be efficient and practical to implement in hardware [3, 13]. This method is shown as the Binary Algorithm:

<u>Binary Algorithm</u>
Define k: number of bits in n and ni: the ith bit of n
Input: P (a point on the elliptic curve).
Output: Q = nP (another point on the elliptic curve).

1. if nk-1 = 1, then Q:=P else Q:=0;
2. for i = k-2 down to 0;
3. { Q := Q +Q ;
4. if ni = 1 then Q:= Q +P ; }
5. return Q;

Basically, the binary algorithm scans the bits of n and doubles the point Q k-times. Whenever, a particular bit of n is found to be one, an extra computation of point addition (Q+P) is needed. Every point addition or point doubling operation requires the three modulo GF(p) operations of inversion, multiplication, and addition/subtraction as presented earlier in Section 2.1.

3. PROJECTIVE COORDINATES

Projective coordinates are used to eliminate the need for performing the lengthy inversion as in the crypto processors in [12, 15]. For elliptic curve defined over GF(p), two different forms of formulae are available [3, 24] for point addition and doubling. One form projects $(x,y)=(X/Z^2,Y/Z^3)$ [3], while the second projects (x,y)=(X/Z,Y/Z) [24].

The two procedures for projective point addition of P+Q (two elliptic curve points) are shown below:

$$P=(X_1,Y_1,Z_1);Q=(X_2,Y_2,Z_2);P+Q=(X_3,Y_3,Z_3);\ where\ P \neq \pm Q$$

$(x,y)=(X/Z^2,Y/Z^3) \Rightarrow (X,Y,Z)$		$(x,y)=(X/Z,Y/Z) \Rightarrow (X,Y,Z)$	
$\lambda_1 = X_1Z_2^2$	2M	$\lambda_1 = X_1Z_2$	1M
$\lambda_2 = X_2Z_1^2$	2M	$\lambda_2 = X_2Z_1$	1M
$\lambda_3 = \lambda_1 - \lambda_2$		$\lambda_3 = \lambda_2 - \lambda_1$	
$\lambda_4 = Y_1Z_2^3$	2M	$\lambda_4 = Y_1Z_2$	1M
$\lambda_5 = Y_2Z_1^3$	2M	$\lambda_5 = Y_2Z_1$	1M
$\lambda_6 = \lambda_4 - \lambda_5$		$\lambda_6 = \lambda_5 - \lambda_4$	
$\lambda_7 = \lambda_1 + \lambda_2$		$\lambda_7 = \lambda_1 + \lambda_2$	
$\lambda_8 = \lambda_4 + \lambda_5$		$\lambda_8 =\lambda_6^2 Z_1Z_2-\lambda_3^2\lambda_7$	5M
$Z_3 = Z_1Z_2\lambda_3$	2M	$Z_3 = Z_1Z_2\lambda_3^3$	2M
$X_3 = \lambda_6^2 - \lambda_7\lambda_3^2$	3M	$X_3 = \lambda_8\lambda_3$	1M
$\lambda_9 = \lambda_7\lambda_3^2 - 2X_3$		$\lambda_9 = \lambda_3^2 X_1Z_2 - \lambda_8$	1M
$Y_3 = (\lambda_9\lambda_6 - \lambda_8\lambda_3^3)/2$	3M	$Y_3 = \lambda_9\lambda_6 - \lambda_3^3 Y_1Z_2$	2M
	-----		-----
	16 M		**15M**

Note that the 16M and 15M represent the total number of multiplication operations (multiplications count) for each procedure, respectively.

Similarly, the two formulae and their multiplication operation count for projective point doubling are shown below:

$$P = (X_1,Y_1,Z_1);\ P+P = (X_3,Y_3,Z_3)$$

$(x,y)=(X/Z^2, Y/Z^3) \Rightarrow (X,Y,Z)$		$(x, y) = (X/Z, Y/Z) \Rightarrow (X,Y,Z)$	
$\lambda_1 = 3X_1^2 + aZ_1^4$	4M	$\lambda_1 = 3X_1^2 + aZ_1^2$	2M
$Z_3 = 2Y_1Z_1$	1M	$\lambda_2 = Y_1Z_1$	1M
$\lambda_2 = 4X_1 Y_1^2$	2M	$\lambda_3 = X_1Y_1\lambda_2$	2M
$X_3 = \lambda_1^2 - 2\lambda_2$	1M	$\lambda_4 = \lambda_1^2 - 8\lambda_3$	1M
$\lambda_3 = 8Y_1^4$	1M	$X_3 = 2\lambda_4\lambda_2$	1M
$\lambda_4 = \lambda_2 - 2X_3$		$Y_3=\lambda_1(4\lambda_3-\lambda_4)-8(Y_1\lambda_2)^2$	3M
$Y_3 = \lambda_1\lambda_4 - \lambda_3$	1M	$Z_3 = 8\lambda_2^3$	2M
	------		-----
	10M		**12M**

The squaring calculation over GF(p) is considered similar to the multiplication computation. They are both noted as M (multiplication). Here the time of addition and subtraction are ignored since they are negligible compared to multiplication [3]. Since the number of projective point additions is taken to be, on an average, half the number of bits, it can be clearly seen form the above tables that the projective coordinate $(x,y) = (X/Z^2,Y/Z^3)$ has on the average 18 multiplication operations, while the projection $(x,y) = (X/Z,Y/Z)$ has on the average 19.5 multiplications. Considering the worst case scenario of having the number of point additions similar to the number of bits, the projective coordinate $(x,y) = (X/Z^2,Y/Z^3)$ has 26 multiplication operations, whereas the projection $(x,y) = (X/Z,Y/Z)$ has 27 multiplications. Clearly, the projective coordinate $(x,y) = (X/Z^2,Y/Z^3)$ would be the projection of choice for sequential implementation, as summarized in Table 1.

Procedure of Projecting	Average Number of Multiplication Cycles	Worst Number of Multiplication Cycles
(x,y) to $(X/Z^2,Y/Z^3)$	18	26
(x,y) to $(X/Z,Y/Z)$	19.5	27

TABLE 1: Comparison Between The Different Projective Coordinate Assuming Single Multiplier (Sequential Implementations)

4. PARALLEL MULTIPLIERS & PREFERRED PROJECTIVE COORDINATES

Our basic motivation in this research is gained by taking advantage of the parallelism that exists in the ECC and its projective coordinate operations. The two forms of projecting procedures (x,y) = (X/Z^2,Y/Z^3) and (x,y) = (X/Z,Y/Z) for projective point addition (P+Q) and projective point doubling (P+P), described in the previous section is studied assuming the flexible possibility of having multiple parallel multipliers in different forms. In principle, the architectures to be considered can operate both coordinate systems. The study assumes having different architectures with their difference in the available number of parallel multipliers they have, as shown in Figure 1. Every architecture with its certain number of multipliers will study the speed difference due to running the two projective coordinates computations. This will conclude the efficient choice of projective coordinate to be adopted for this hardware with this specific number of multipliers. This section will study the algorithms of both projective coordinates and their best map of data dependency based on parallel multipliers which, in reality, will affect the controlling unit within the architecture to make it efficient. Note that the detailed multiplier design will not change the overall hardware design nor the comparisons results, therefore this level of details are not considered in the focus of this study.

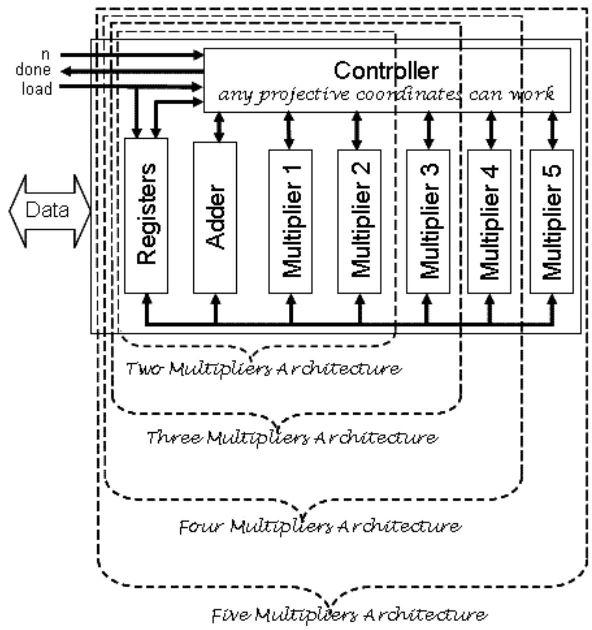


FIGURE 1: General Architecture Showing All Different Designs of This Study - Based On Its Number of Parallel Multipliers

We found that the architectures does not show speed improvement due to parallelization when higher number of multipliers are used, i.e. when 5 or more are involved. It is found that both projective coordinate forms can be parallelized giving improving results to the maximum possibility using four multipliers. This can be observed from the details of the following subsections.

4.1 Parallelizing Multiplications of the Standard Coordinates

The standard projection of (x,y) = (X/Z,Y/Z) is assumed to be running on the different architectures of Figure 1. It will be tested differently involving 2, 3, 4, and 5 multipliers processed in parallel. Figures 2 and 3 highlight the dependency within the procedures when two parallel multipliers are considered. The figures detail the different critical path stages, hence different number of multiplication cycles needed for the operations. It can observed that the projection (x,y) = (X/Z,Y/Z) needs 8 multiplications for point addition and 6 for point doubling. Using the common assumption of the number of point additions to be half the number of bits, we can assume the average number as 4 additions and 6 doubling resulting 10 multiplications. Allowing for the worst case of having the number of point additions to be equal to number of bits, the projective coordinate will need 14 multiplications.

Consider the architecture of Figure 1 with three multipliers. The standard projection of (x,y)=(X/Z,Y/Z) is need of 5 multiplications for point adding (Figure 4) and 4 for point doubling (Figure 5). This will make the worst case number of multiplications as 9, whereas it is 6.5 as average scenario.

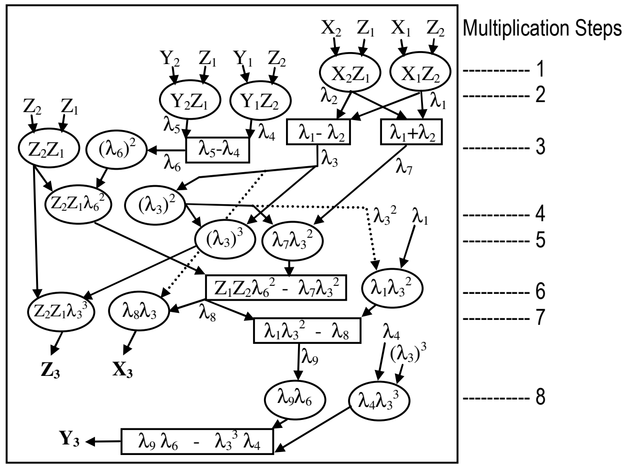


FIGURE 2: Addition Data Flow Diagram For Projection of *(X,Y)* To *(X/Z,Y/Z)* Using 2 Parallel Multipliers

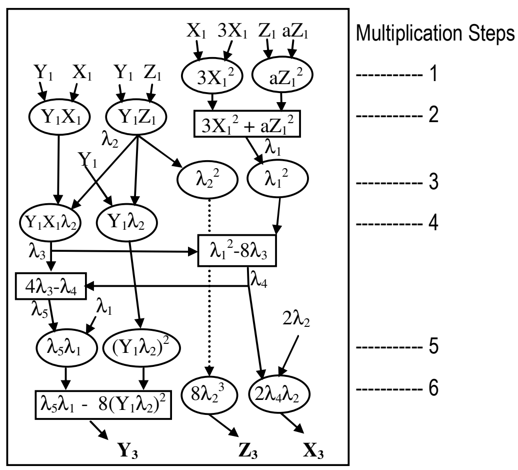


FIGURE 3: Doubling Data Flow Diagram For Projection of *(X,Y)* To *(X/Z,Y/Z)* Using 2 Parallel Multipliers

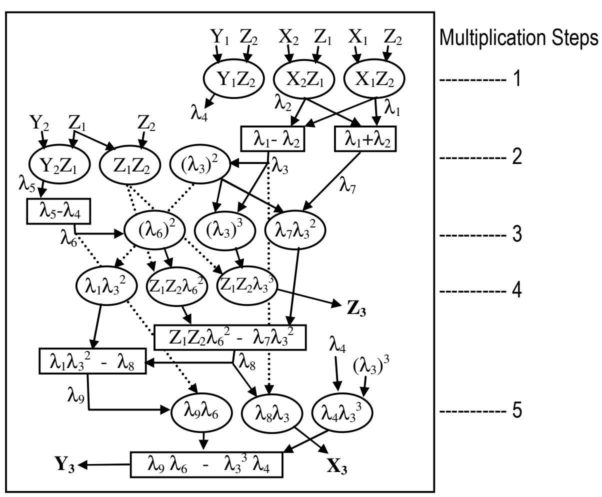


FIGURE 4: Addition Data Flow Diagram For Projection of (X,Y) To (X/Z,Y/Z) Using 3 Parallel Multipliers

The standard projection of (x,y) = (X/Z,Y/Z) can operate with 4 multipliers as presented earlier in [26]. This hardware is providing results after 4 multiplication steps for point adding and after 3 steps for point doubling as in Figures 6, and 7, respectively. This design worst case number of multiplications steps is 7, while its average is 5 running this projective system.

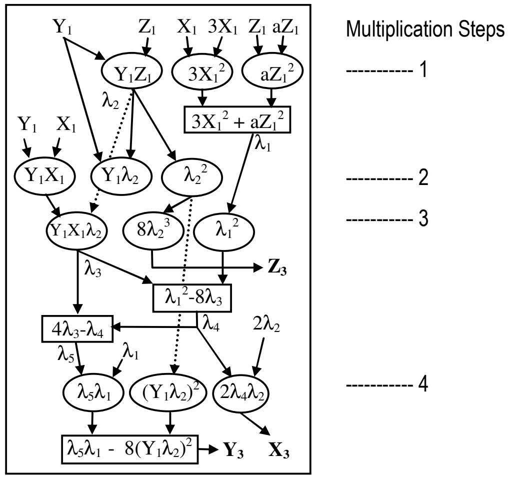


FIGURE 5: Doubling Data Flow Diagram For Projection of *(X,Y)* To *(X/Z,Y/Z)* Using 3 Parallel Multipliers

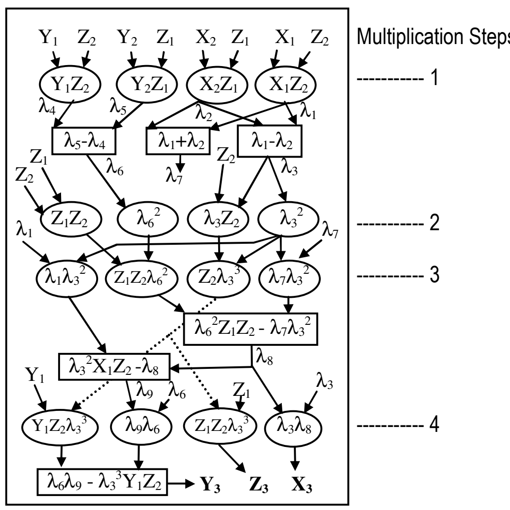


FIGURE 6: Addition Data Flow Diagram For Projection of *(X,Y)* To *(X/Z,Y/Z)* Using 4 Parallel Multipliers

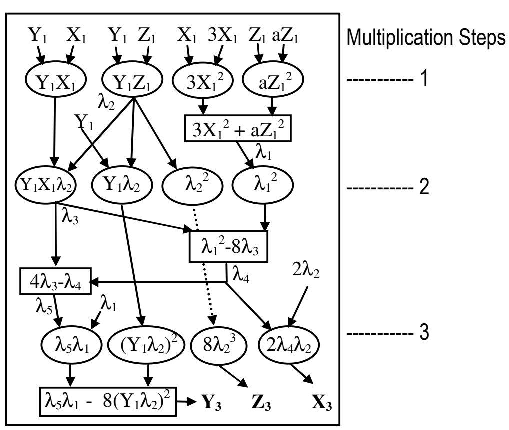


FIGURE 7: Doubling Data Flow Diagram For Projection of *(X,Y)* To *(X/Z,Y/Z)* Using 4 Parallel Multipliers

The last architecture study for projecting (x,y) to (X/Z,Y/Z) is assuming processing utilizing 5 parallel multipliers. As shown in Figure 8, although this hardware is having higher capability of parallelization, the number of multiplication steps cannot be reduced more, i.e. more than the architecture of 4 multipliers. This made the option of five multipliers not recommended for this coordinate system as well as the Jacobian system as will be shown in the next subsection.

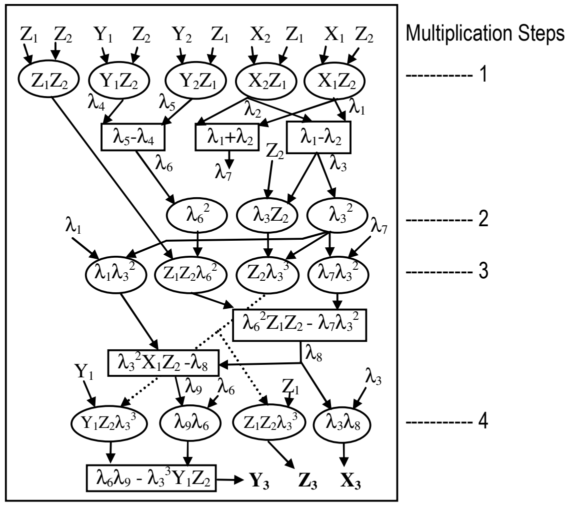


FIGURE 8: Addition Data Flow Diagram For Projection of *(X,Y)* To *(X/Z,Y/Z)* Using 5 Parallel Multipliers

4.2 Parallelizing Multiplications of the Jacobian Coordinates

The other system to be studied for operation on the architectures of Figure 1 is the Jacobian projective coordinate of (x,y) = $(X/Z^2,Y/Z^3)$. When Jacobian procedure is mapped on two multipliers hardware, the point addition operation is in need of 8 multiplication steps and the point doubling needs 5 steps as in Figures 9 and 10, respectively. The average number of multiplication steps is considered 9 multiplication cycles, assuming the common theory of the number of point additions to be half the number of bits, as the Binary algorithm of Section 2.3. The longest case possible is when the number of point additions equal to number of bits making the projective coordinate (x,y) = $(X/Z^2,Y/Z^3)$ in need for 13 multiplication operation steps.

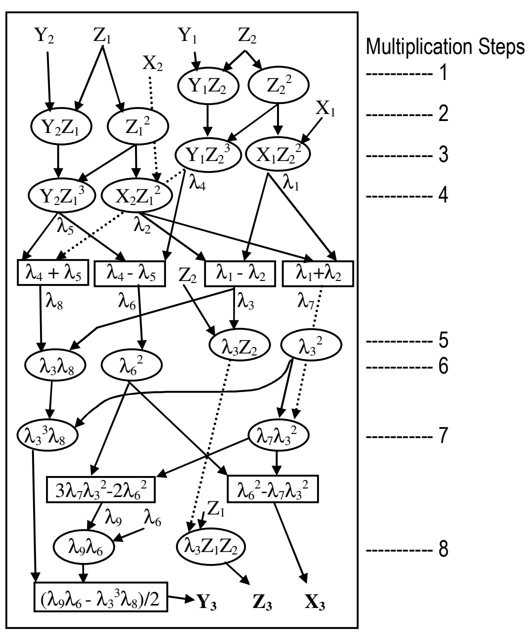


FIGURE 9: Addition Data Flow Diagram For Projection of *(X,Y)* To *$(X/Z^2,Y/Z^3)$* Using 2 Parallel Multipliers

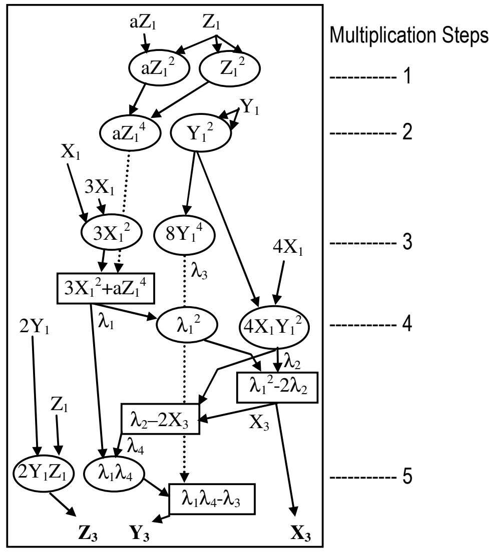


FIGURE 10: Doubling Data Flow Diagram For Projection of *(X,Y)* To $(X/Z^2,Y/Z^3)$ Using 2 Parallel Multipliers

The Jocbian projection of $(x,y) = (X/Z^2,Y/Z^3)$ operate on the 3 multipliers architecture of Figure 1 as presented in Figure 11 for point addition and Figure 12 for point doubling. This hardware can provide outcome after 6 multiplication steps for point adding and after 4 steps for point doubling. This model worst case number of multiplications steps is 10, while its average is 7 operating this projective coordinate procedure.

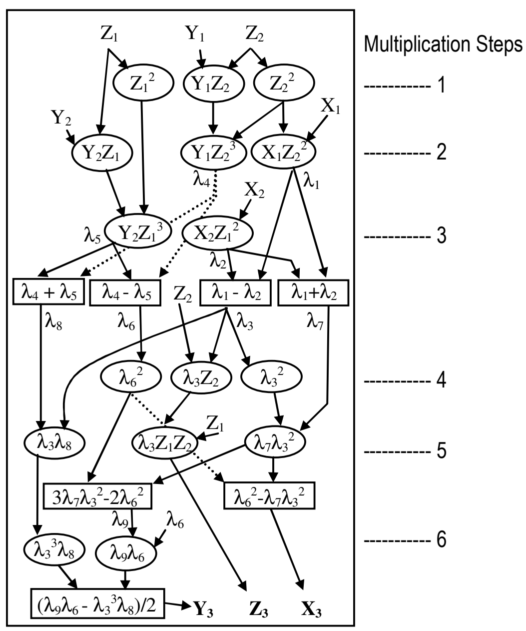


FIGURE 11: Addition Data Flow Diagram For Projection of *(X,Y)* To $(X/Z^2,Y/Z^3)$ Using 3 Parallel Multipliers

Figure 13 and Figure 14 is showing the data flow when running the projective coordinate $(x,y) = (X/Z^2,Y/Z^3)$ on hardware with 4 multipliers as implemented earlier in [26]. On the average, the design needs 6.5 multiplication cycles. Allowing for the worst case of having the number of point additions to be equal to number of bits, the projective coordinate $(x,y) = (X/Z^2,Y/Z^3)$ need 9 multiplication operations.

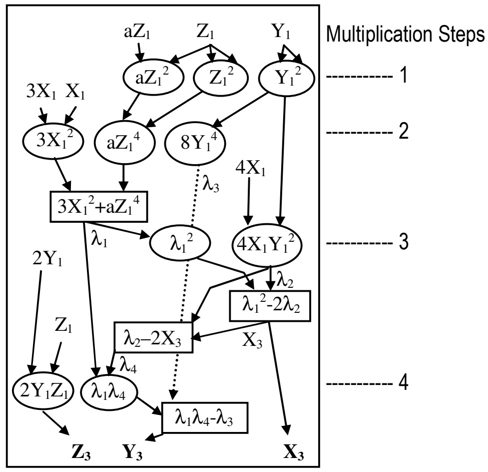


FIGURE 12: Doubling Data Flow Diagram For Projection of *(X,Y)* To *$(X/Z^2,Y/Z^3)$* Using 3 Parallel Multipliers

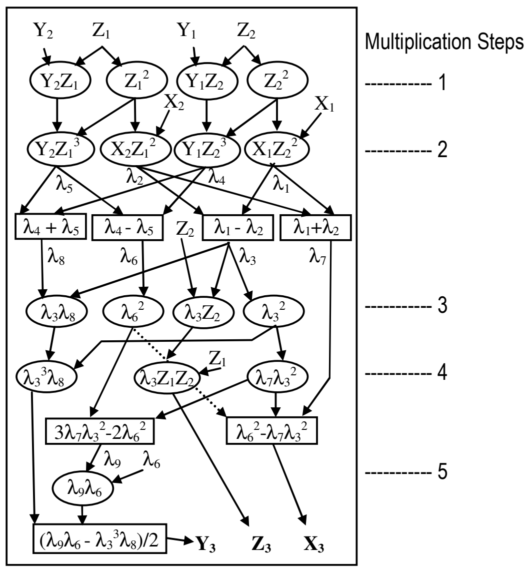


FIGURE 13: Addition Data Flow Diagram For Projection of *(X,Y)* To *$(X/Z^2,Y/Z^3)$* Using 4 Parallel Multipliers

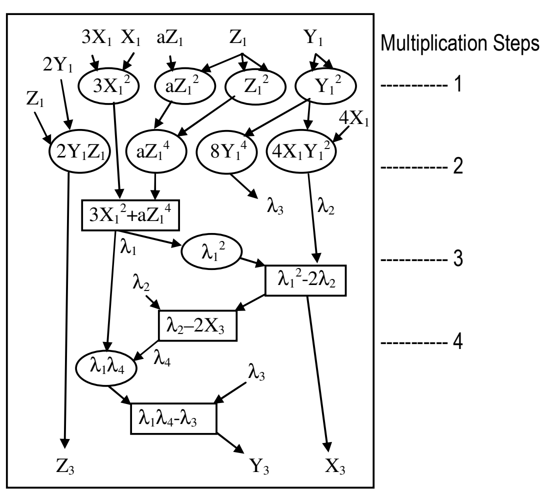


FIGURE 14: Doubling Data Flow Diagram For Projection of *(X,Y)* To *$(X/Z^2,Y/Z^3)$* Using 4 Parallel Multipliers

Similar to the study of mapping the standard projective coordinates on the architecture of 5 multipliers, the Jacobian projection of (x,y) to $(X/Z^2,Y/Z^3)$ is mapped on 5 parallel multipliers in Figure 15. This design is ideally having more capability of parallelization, however, the number of multiplication steps cannot be reduced more, i.e. more than the architecture of 4 multipliers. This is found for both coordinate systems making the 5 multipliers hardware not recommended.

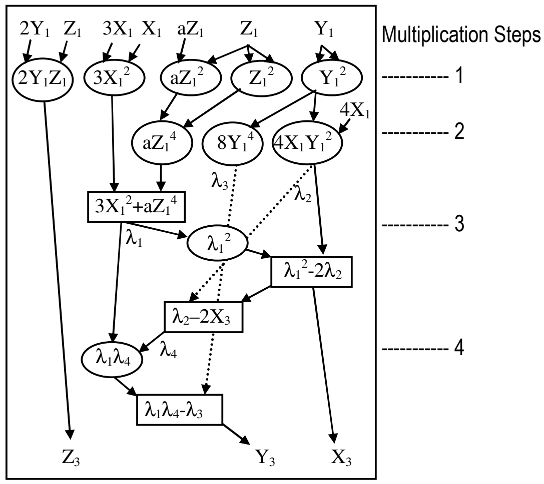


FIGURE 15: Doubling Data Flow Diagram For Projection of *(X,Y)* To $(X/Z^2,Y/Z^3)$ Using 5 Parallel Multipliers

4.3 Preferred – Efficient- Projective Coordinate

Both projection systems, i.e. Standard and Jacobian, can operate on all architectures of Figure 1. However, every architecture prefers running a specific projection procedure based on the inherent parallelism possible in its multiplication processes. This choice leads to differences in the controller, which maps the operations efficiently as listed in Table 2. Also, the preferred procedures lead to better utilization of multipliers in that specific hardware as will be described later in this section.

Procedure of Projecting	# Parallel Multipliers	Number of Multiplication Cycles		Multipliers Utilization		Preferred (Efficient) Projective Coordinate
		Average	Worst	Average	Worst	
(x,y) to $(X/Z^2,Y/Z^3)$	1	8+10=18	16+10=26	100%	100%	√
(x,y) to $(X/Z,Y/Z)$		7.5+12=19.5	15+12=27	100%	100%	
(x,y) to $(X/Z^2,Y/Z^3)$	2	4+5=9	8+5=13	100%	100%	√
(x,y) to $(X/Z,Y/Z)$		4+6=10	8+6=14	98%	96%	
(x,y) to $(X/Z^2,Y/Z^3)$	3	3+4=7	6+4=10	86%	87%	
(x,y) to $(X/Z,Y/Z)$		2.5+4=6.5	5+4=9	100%	100%	√
(x,y) to $(X/Z^2,Y/Z^3)$	4	2.5+4=6.5	5+4=9	69%	72%	
(x,y) to $(X/Z,Y/Z)$		2+3=5	4+3=7	100%	100%	√
(x,y) to $(X/Z^2,Y/Z^3)$	5	2.5+4=6.5	5+4=9	55%	58%	
(x,y) to $(X/Z,Y/Z)$		2+3=5	4+3=7	80%	80%	√

TABLE 2: Preferred - Efficient - Projective Coordinates and Utilization According to Number of Parallel Multipliers

It is found that the Jacobian projection (x,y) = $(X/Z^2,Y/Z^3)$ leads to less number of multiplication steps for one and two multipliers hardware, as in Table 2. It is to be noted that this projection mapping is providing 100% utilization of the multipliers, which is the efficient hardware usage.

When the number of multipliers exceeds 2, i.e. 3 and 4, the standard projection of (x,y) = (X/Z,Y/Z) gives less number of parallel multiplication steps, which would be the projection of choice for our

implementation. Also, this system is utilizing is 100% hardware, using the four multipliers in all multiplication cycles, as brifed in Table 2, which is not the case of the projection (x,y) = (X/Z^2,Y/Z^3).
If the design is made of 5 multipliers, the speed will not change than the 4 multipliers preferring the standard projection system. Interestingly, it also shows hardware utilization of 80%, which is better than the Jacobian coordinate system that is showing 50% ~ 58% utilization. This made the decision to avoid the 5 multiplier architecture from making it as an option of choice for both Standard and Jacobian coordinate systems.

5. ARCHITECTURES AND EFFICIENCY DECISIONS

Several cryptographic architectures have been proposed in the literature [11, 12, 15, 18]. The conventional approach used in the design of these processors is to adopt serial computations at both the algorithmic level by using a single multiplier, as well as at the arithmetic level by using a serial multiplier. The reason for serial multiplier and sequential operation is that they lead to the lowest area for large word lengths, which is needed for secure encryption (i.e. > 160 bits [3]). The above approach reduces area at the expense of speed. The new architectures study proposed in this paper have multi parallel multipliers, an adder, registers and a controller. The design is a straightforward implementation of the dependency graphs (Section 4) based on the number of multipliers needed and the efficient projective coordinate selected. The designs controller is constructed of a finite state machine to direct the flow of data to conduct the required projective point operation depending on the binary algorithm described previously in Section 3. This section will also consider the comparison between these architecture with respect to their cost in relation to the area and speed (time). The time will be factor of the number of bits to be computed as needed by the crypto application. The improvement in our crypto-processor is focusing on the parallelism described in Section 4 and not on the basic GF(p) multiplier and adder, nor hardware details.

To mention briefly about the suggested multipliers, the designs proposed in [14, 15] use multiplier hardware that is fixed to number of bits they are intended for, if the number of bits are needed to be increased for some crypto application, the complete processor is to be replaced. Furthermore, if the number of bits is much less than the intention of the hardware design, the unnecessary bits will be considered as zeros but included in the computation, causing the same delay exactly as if all bits are essential. These weaknesses motivated the recommendation to choose adopting special scalable multipliers instead of conventional as detailed in the design in [26].

In this study, the number of multipliers is used as the area factor as an assumption for comparison reasons and the timing will assume 160, 256 and 512 bits for crypto calculations, which are the common number of bits needed by most applications [28, 29, 31]. The area factor and timing average estimate will be multiplied together to generate different cost figures. These cost figures are just simple figure of merit values to be used for evaluation reasons. For example, the cost AT (AT = A×T), assumes that time and area is having similar balanced importance to the application. When timing is more important, the cost figure of merit AT is assumed to be further multiplied by time T making it ATT (ATT = A×T×T). These studies are similar to the cost studies provided in [26]. On similar concept but with allowing for the application to have more importance to area than time, we included in this study the cost AAT, where the area is squared multiplied by the timing once. This new AAT cost is believed to be needed for applications with very limited hardware area such as smart cards and small mobile devices, where area is more important than speed.

Considering the different architectures with different number of multipliers as described in section 4.3 before, we start our focus on the design of two multipliers. It prefers the Jacobian coordinates to form its controller hardware, as shown in Figure 16. The average timing of the process is in need of 9 cycles per bit making the total timing estimation for 160 bits crypto computations as T = 1,440 cycles. Assuming the area factor is A = 2 multipliers, the cost: AT = 2,880; the cost ATT = 4,147,200; and the cost AAT = 5,760. Again, to be explicit, these cost (figure of merit) results does not have a direct meaning in its value except to compare the different designs efficiency and help making proper decision. Similarly, the timing of running this hardware on 256 bits is T = 2,304 cycle. The costs will be: AT = 4,608; ATT = 10,616,832; and AAT = 9,216. The costs of this hardware computing 512 bits are AT = 9,216; ATT = 42,467,328; and AAT = 18,432.

When the architecture is having three or four multipliers, the preferred controller state machine should run the standard projective coordinates, as mentioned in Table 2 earlier. The hardware with three multipliers (Figure 17) runs 160 bits crypto applications with average estimated timing $T = 160 \times 6.5 = 1{,}040$ cycles. This hardware cost $AT = 3 \times 1040 = 3{,}120$; $ATT = 3{,}244{,}800$ and $AAT = 9{,}360$. The timing when running 256 bits will be changed to $T = 256 \times 6.5 = 1{,}664$ cycles. The cost is changed to $AT = 3 \times 1664 = 4{,}992$; $ATT = 3 \times 1664 \times 1664 = 8{,}306{,}688$; and $AAT = 14{,}976$. The costs values of this hardware when used for 512 bits computation adjusted to $AT = 9{,}984$; $ATT = 33{,}226{,}752$; and $AAT = 29{,}952$.

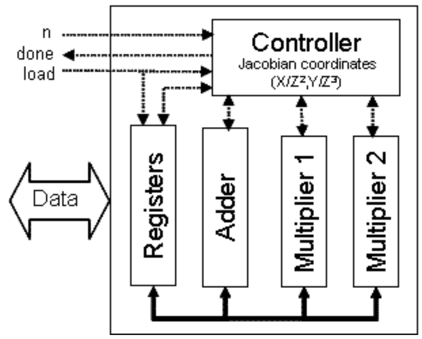


FIGURE 16: Elliptic Curve Processor Architecture Using 2 Multipliers

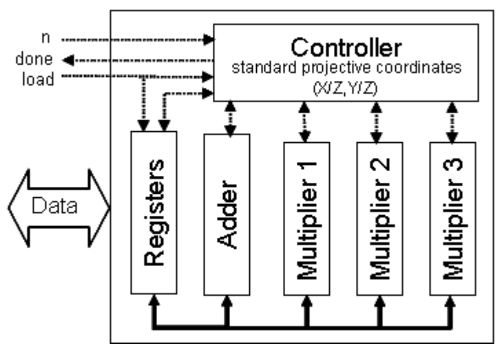


FIGURE 17: Elliptic Curve Processor Architecture Using 3 Multipliers

The four multipliers architecture running standard projective coordinate system, as Figure 18, needs 5 cycles per bit (on average estimate timing). When the total bits to be processed is 160, the timing shows $T = 160 \times 5 = 800$ cycles. The area factor $A = 4$, making the cost $AT = 3{,}200$; $ATT = 2{,}560{,}000$; and $AAT = 12{,}800$. If the number of bits is 256, the timing is modified to $T = 256 \times 5 = 1{,}280$; the cost will be $AT = 5{,}120$; $ATT = 6{,}553{,}600$; and $AAT = 20{,}480$. For 512 bits computations, the costs are found to be $AT = 10{,}240$; $ATT = 26{,}214{,}400$; and $AAT = 40{,}960$.

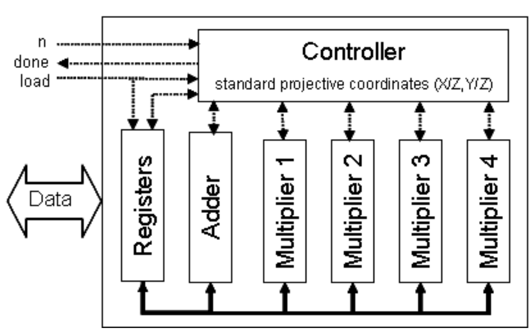


FIGURE 18: Elliptic Curve Processor Architecture Using 4 Multipliers

To complete the study, the cost is also computed for the architecture with five multipliers although it is already estimated as inefficient design (Section 4.3). The timing for 160 bits is found T = 160 × 5 = 800 cycles, similar to the four multiplier hardware. However, the cost AT = 5 × 800 = 4,000; ATT = 5 × 800 × 800 = 3,200,000; and AAT = 20,000. For the 256 bits, the timing T = 256 × 5 = 1,280 cycles. The cost will be AT = 5 × 1280 = 6,400; ATT = 5 × 1280 × 1280 = 8,192,000; and AAT = 32,000. If 512 bits are running on this hardware, the cost evaluation values are AT = 12,800; ATT = 32,768,000; and AAT = 64,000. To summarize this area time parameters effect for the different architectures, all cost values are listed in Table 3.

Number of Multipliers (A)		1	2	3	4	5
Avg. Timing per Bit		18	9	6.5	5	5
Crypto Applications Bits = 160	Total Time Number of Cycles (T)	2880	1440	1040	800	800
	Cost (AT)	2880	2880	3120	3200	4000
	Cost (ATT)	8294400	4147200	3244800	2560000	3200000
	Cost (AAT)	2880	5760	9360	12800	20000
Crypto Applications Bits = 256	Total Time Number of Cycles (T)	4608	2304	1664	1280	1280
	Cost (AT)	4608	4608	4992	5120	6400
	Cost (ATT)	21233664	10616832	8306688	6553600	8192000
	Cost (AAT)	4608	9216	14976	20480	32000
Crypto Applications Bits = 512	Total Time Number of Cycles (T)	9216	4608	3328	2560	2560
	Cost (AT)	9216	9216	9984	10240	12800
	Cost (ATT)	84934656	42467328	33226752	26214400	32768000
	Cost (AAT)	9216	18432	29952	40960	64000

TABLE 3: Summary of Cost for the Different Architectures

Considering the different costs of all architectures makes the decision to choose specific hardware more efficient. For example, if the hardware area and speed are having the same level of significance, Figure 19 shows that the architectures of one or two multipliers can both give similar costs. This case is verified for all different number of bits used.

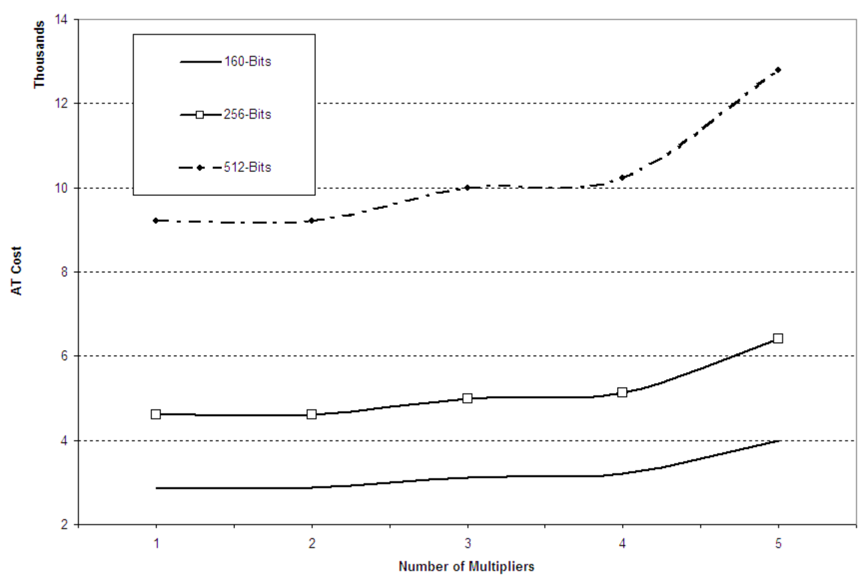


FIGURE 19: AT Cost Comparison of Preferred Architectures

When the hardware area is less important than the speed, Figure 20 makes the selection for four multipliers design. This is valid for all number of bits; in fact, it can be seen more clearly as the number increase to 512 bits.

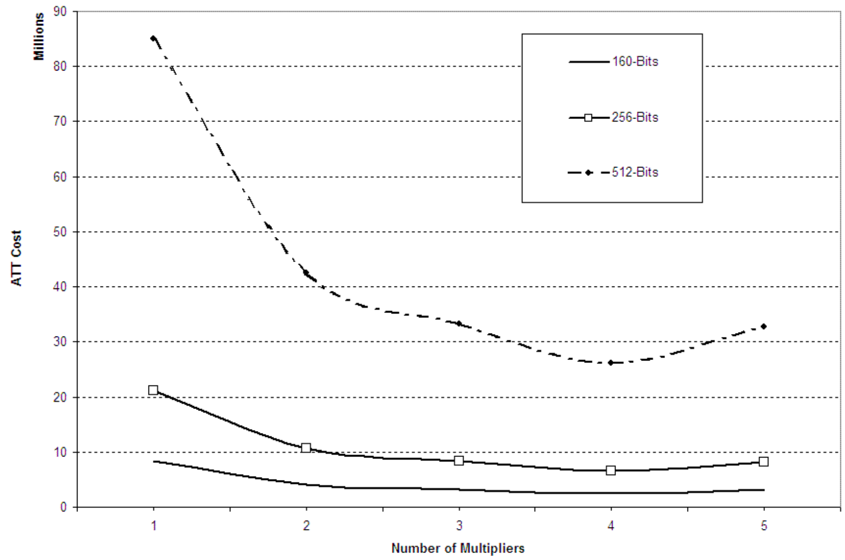


FIGURE 20: ATT Cost Comparison of Preferred Architectures

Interestingly, the choice of 4 multiplier hardware is also applicable when the crypto hardware is assuming importance of area more than speed, as case shown in Figure 21. This indicates that whenever time and speed are not having the same importance, four multipliers hardware running standard projective coordinates will be the efficient architecture to use.

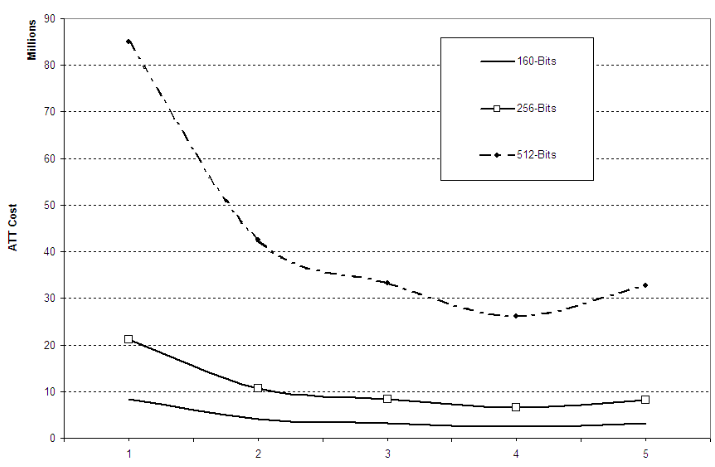


FIGURE 21: AAT Cost Comparison of Preferred Architectures

6. CONCLUSION

This paper presented a modeling investigation for efficient architectures used in elliptic curve cryptography (ECC) computations. We present different architectural study of designs having two, three, four and five multipliers operating in parallel. The original idea is based on the utilization of the inherited parallelism of multiplication steps in the procedures adopted. The work assumes that the ECC inverse operations are converted into consecutive multiplication steps through projective coordinates where two well known forms of procedures for projective coordinates are considered, i.e Jacobian and standard projective coordinates. Comparing the two projective forms running on a single multiplier hardware shows that projecting (x,y) to $(X/Z^2,Y/Z^3)$ (Jacobian coordinates) requires less number of multiplications than projecting into (X/Z,Y/Z) (standard coordinates). Standard projection uses one less multiplication operation in adding two different elliptic points, however, it uses two more multiplication operations in doubling an elliptic point. This made the normal choice for sequential implementation, i.e. using a single multiplier, that projecting (x,y) into $(X/Z^2,Y/Z^3)$ has always been the candidate of choice for implementing ECC since it has the minimum number of multiplication operations.

Although the proposed architectures can handle the algorithmic procedures of both projective coordinate forms, the analysis of the critical paths of both projective coordinates indicates that for parallel multipliers hardware, projecting (x,y) to (X/Z,Y/Z) requires less number of cycles (faster hardware) and better utilization than projecting (x,y) to $(X/Z^2,Y/Z^3)$ whenever the number of multipliers are more than two, which gives the choice for performance boost up.

The presented work also involved a cost comparison that refers to the application main concentration. This cost study is formed by relating between the area and speed for every architecture. The analysis proven the efficiency of designs involving one or two multipliers when both area and speed factors are having similar importance to the application. However, this study recommended the preference of hardware with 4 multipliers whenever the application is having area or speed (one of the cost factors) as more important. The attraction of this work is that using the proposed architecture with projections of (x,y) to (X/Z,Y/Z) is leading to the best performance, utilizing the maximum inherited parallelism of the projective coordinate arithmetic.

Furthermore, this work can be a seed for software implementations of the same ECC system on currently available multi-core general purpose processors (multi-core processors). Most of this study can be tuned for parallel programming assuming every multiplier is in a different core in the processor. The program to be written need to consider this issue from early ahead to help the compilers in their parallelization tasks. The number of cores to be used can take into consideration the different software programs running simultaneously, which can also be dynamically changing based on the application need.

ACKNOWLEDGMENTS

The author would like to thank Umm Al-Qura University and King Fahd University of Petroleum and Minerals for their unlimited support for all research work.

REFERENCES

[1] V. S. Miller, “Use of Elliptic Curves in Cryptography”, *Proceedings of Advances in Cryptology (Crypto),* (1986), p. 417–426.

[2] N. Koblitz, “Elliptic Curve Cryptosystems”, *Math. Computing*, **48** (1987), p. 203–209.

[3] Blake, Seroussi, and Smart, Elliptic Curves in Cryptography, Cambridge University Press: New York, 1999.

[4] G. V. S. Raju, R. Akbani, “Elliptic Curve Cryptosystem and its Applications”, *IEEE International Conference on Systems, Man and Cybernetics*, **2** (2003), p.1540 – 1543.

[5] Paar, Fleischmann, and Soria-Rodriguez, “Fast Arithmetic for Public-Key Algorithms in Galois Fields with Composite Exponents”, *IEEE Transactions on Computers*, **48**:10 (1999).

[6] R. L. Rivest, A. Shamir, and L. Adleman, “A Method for Obtaining Digital Signatures and Public-key Cryptosystems”, *Communications of the ACM*, **21**:2 (1978), p. 120–126.

[7] IEEE P1363, http://grouper.ieee.org/groups/1363

[8] The ATM Forum, http://www.atmforum.com/meetings/rich_bios. html

[9] The Internet Engineering Task Force, http://www.ietf.cnri.reston.va.us

[10] Chung, Sim, Lee, "Fast Implementation of Elliptic Curve Defined over GF(p^m) on CalmRISC with MAC2424 Coprocessor", *Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, (2000).

[11] Okada, Torii, Itoh, Takenaka, "Implementation of Elliptic Curve Cryptographic Coprocessor over GF(2^m) on an FPGA", *Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, (2000).

[12] Orlando and Paar, "A High-Performance Reconfigurable Elliptic Curve Processor for GF(2^m)", *Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, (2000).

[13] Hankerson, Hernandez, and Menezes, "Software Implementation of Elliptic Curve Cryptography Over Binary Fields", *Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, (2000).

[14] G. A. Orton and others, "VLSI implementation of public-key encryption algorithms", *Advances in Cryptology (CRYPTO)*, **263 (**1986), p. 277-301.

[15] Orlando and Paar, "A scalable GF(p) elliptic curve processor architecture for programmable hardware", *Cryptographic Hardware and Embedded Systems (CHES)*, (2001).

[16] Royo, Moran, and Lopez, "Design and implementation of a coprocessor for cryptography applications", *European Design and Test Conference Proceedings*, (1997), p. 213–217.

[17] Agnew, Mullin, and Vanstone, "An Implementation of Elliptic Curve Cryptosystems Over F_2^{155}", *IEEE Journal on Selected Areas in Communications*, **11**:5 (1993), p. 804–813.

[18] Sıddıka Berna Ors and others, "Hardware Implementation of an Elliptic Curve Processor over GF(p)", *Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP)*, (2003), p. 433 – 443.

[19] G.B. Agnew, R.C. Mullin, and S.A.Vanstone, "An implementation of elliptic curve cryptosystems over $F_2$155", *IEEE Journal on Selected Areas in Communications*, **11**:5 (1993), p.804-813.

[20] Chi Huang, Jimnei Lai, Junyan Ren, and Qianling Zhang, "Scalable Elliptic Curve Encryption Processor for Portable Application", *Proceedings of the 5th International Conference on ASIC*, **2** (2003), p. 1312-1316.

[21] J. R. Michener and S. D., "Mohan, Internet Watch: Clothing the E-Emperor, Computer – Innovative Technology for Computer Professionals", *IEEE Computer Society*, **34**:9 (2001), p. 116-118.

[22] Adnan Abdul-Aziz Gutub, A. F. Tenca, and C. K. Koc, "Scalable VLSI architecture for GF(p) Montgomery modular inverse computation", *IEEE Computer Society Annual Symposium on VLSI*, (2002), p. 53-58.

[23] A. F. Tenca and C. K. Koc, "A Scalable Architecture for Modular Multiplication Based on Montgomery's Algorithm", *IEEE Transactions on Computers*, **52**:9 (2003), p. 1215-1221.

[24] A. Miyaji, "Elliptic Curves over F_P Suitable for Cryptosystems", *Advances in cryptology-AUSCRUPT'92*, Australia, (1992).

[25] Mentor Graphics Co., http://www.mentor. com/partners/hep/AsicDesignKit/dsheet/ami05databook.html, ASIC Design Kit.

[26] Adnan Gutub, "Efficient Utilization of Scalable Multipliers in Parallel to Compute GF(p) Elliptic Curve Cryptographic Operations", *Kuwait Journal of Science & Engineering (KJSE)*, Vol . 34, No. 2, Pages: 165-182, December 2007.

[27] Daniel J. Bernstein1 and Tanja Lange, "Faster Addition and Doubling On Elliptic Curves," *Springer Berlin /Heidelberg, Supported in Part by The European Commission Through The 1st Programme,* Vol. 4833/2008, November 05, 2007.

[28] Adnan Gutub, Mohammad Ibrahim, and Turki Al-Somani, "Parallelizing GF(P) Elliptic Curve Cryptography Computations for Security and Speed", *IEEE International Symposium on Signal Processing and its Applications in conjunction with the International Conference on Information*

Sciences, Signal Processing and their Applications (ISSPA), Sharjah, United Arab Emirates, February 12-15,2007.

[29] Adnan Gutub, "Fast 160-Bits GF(p) Elliptic Curve Crypto Hardware of High-Radix Scalable Multipliers", *International Arab Journal of Information Technology (IAJIT)*, Vol. 3, No. 4, Pages: 342-349, October 2006.

[30] L. Tawalbeh and A. Tenca, "An Algorithm and Hardware Architecture for Integrated Modular Division and Multiplication in GF(P) and GF(2^N)," *IEEE International Conference on Application-Specific Systems,* April 2004.

[31] L. Tawalbeh, "A Novel Unified Algorithm And Hardware Architecture for Integrated Modular Division and Multiplication in GF(P) and GF(2^N) Suitable for Public-Key Cryptography", *Ph.D. Thesis, School of Electrical Engineering and Computer Science,* Oregon State University, October 28, 2004.

CALL FOR PAPERS

Journal: International Journal of Security (IJS)
Volume: 4 **Issue:** 5
ISSN:1985-2320
URL: http://www.cscjournals.org/csc/description.php?JCode=IJS

About IJS

Information Security is an important aspect of protecting the information society from a wide variety of threats. The International Journal of Security (IJS) presents publications and research that builds on computer security and cryptography and also reaches out to other branches of the information sciences. Our aim is to provide research and development results of lasting significance in the theory, design, implementation, analysis, and application of secure computer systems.

IJS provides a platform to computer security experts, practitioners, executives, information security managers, academics, security consultants and graduate students to publish original, innovative and time-critical articles and other information describing research and good practices of important technical work in information security, whether theoretical, applicable, or related to implementation. It is also a platform for the sharing of ideas about the meaning and implications of security and privacy, particularly those with important consequences for the technical community. We welcome contributions towards the precise understanding of security policies through modeling, as well as the design and analysis of mechanisms for enforcing them, and the architectural principles of software and hardware system implementing them.

To build its International reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJS.

IJS List of Topics

The realm of International Journal of Security (IJS) extends, but not limited, to the following:

- Anonymity
- Attacks, security mechanisms, and security service
- Authorisation
- Cellular/wireless/mobile/satellite networks securi
- Public key cryptography and key management
- Anonymity and pseudonymity
- Code security, including mobile code security
- Biometrics
- Authentication
- Confidentiality, privacy, integrity, authenticatio

- Cryptography and cryptanalysis
- Data confidentiality issues
- Data integrity issues
- Data recovery
- Database security
- Denial of service
- Denial of service attacks and countermeasures
- Dependability and reliability
- Design or analysis of security protocols
- Distributed access control
- Distributed and parallel systems security
- Electronic commerce
- Formal security analyses
- Fraudulent usage
- Information flow
- Information hiding and watermarking
- Intellectual property protection
- Intrusion detection
- Key management
- Multicast security
- Network and Internet security
- Network forensics
- Network security performance evaluation
- Non-repudiation
- Peer-to-peer security
- Prevention of traffic analysis
- Privacy protection
- Computer forensics
- Revocation of malicious parties
- Risk assessment and management
- Secure location determination
- Secure PHY/MAC/routing protocols
- Secure routing protocols
- Security group communications
- Security in ad hoc networks
- Security in cellular networks (2G, 2.5G, 3G, B3G,
- Security in communications
- Security in content-delivery networks
- Security in distributed systems
- Security in domain name service
- Security in e-mail
- Security in high-speed networks
- Security in integrated networks
- Security in integrated wireless networks
- Security in internet and WWW
- Security in IP networks
- Security in mobile IP
- Security in optical systems and networks
- Security in peer-to-peer networks
- Security in satellite networks
- Security in sensor networks
- Security in VoIP
- Security in wired and wireless integrated networks
- Security in Wired Networks
- Security in wireless communications
- Security in wireless internet
- Security in wireless LANs (IEEE 802.11 WLAN, WiFi,
- Security in wireless MANs (IEEE 802.16 and WiMAX)
- Security in wireless PANs (Bluetooth and IEEE 802.
- Security policies
- Security specification techniques
- Security standards
- Tradeoff analysis between performance and security
- Trust establishment
- Viruses worms and other malicious code
- WLAN and Bluetooth security

Important Dates

Volume: 4
Issue: 5
Paper Submission: September 30 2010
Author Notification: November 01, 2010
Issue Publication: November / December 2010

CALL FOR EDITORS/REVIEWERS

CSC Journals is in process of appointing Editorial Board Members for ***International Journal of Computer Security (IJS)***. CSC Journals would like to invite interested candidates to join **IJS** network of professionals/researchers for the positions of Editor-in-Chief, Associate Editor-in-Chief, Editorial Board Members and Reviewers.

The invitation encourages interested professionals to contribute into CSC research network by joining as a part of editorial board members and reviewers for scientific peer-reviewed journals. All journals use an online, electronic submission process. The Editor is responsible for the timely and substantive output of the journal, including the solicitation of manuscripts, supervision of the peer review process and the final selection of articles for publication. Responsibilities also include implementing the journal's editorial policies, maintaining high professional standards for published content, ensuring the integrity of the journal, guiding manuscripts through the review process, overseeing revisions, and planning special issues along with the editorial team.

A complete list of journals can be found at http://www.cscjournals.org/csc/byjournal.php. Interested candidates may apply for the following positions through http://www.cscjournals.org/csc/login.php.

Please remember that it is through the effort of volunteers such as yourself that CSC Journals continues to grow and flourish. Your help with reviewing the issues written by prospective authors would be very much appreciated.

Feel free to contact us at coordinator@cscjournals.org if you have any queries.

Contact Information

Computer Science Journals Sdn BhD
M-3-19, Plaza Damas Sri Hartamas
50480, Kuala Lumpur MALAYSIA

Phone: +603 6207 1607
+603 2782 6991
Fax: +603 6207 1697

BRANCH OFFICE 1
Suite 5.04 Level 5, 365 Little Collins Street,
MELBOURNE 3000, Victoria, AUSTRALIA

Fax: +613 8677 1132

BRANCH OFFICE 2
Office no. 8, Saad Arcad, DHA Main Bulevard
Lahore, PAKISTAN

EMAIL SUPPORT
Head CSC Press: coordinator@cscjournals.org
CSC Press: cscpress@cscjournals.org
Info: info@cscjournals.org

COMPUTER SCIENCE JOURNALS SDN BHD
M-3-19, PLAZA DAMAS
SRI HARTAMAS
50480, KUALA LUMPUR
MALAYSIA