2009-06

Investigation into the impacts of migration to emergent NSA Suite B encryption standards

Shu, Jonathan Lee Yee

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

INVESTIGATION INTO THE IMPACTS OF MIGRATION TO EMERGENT NSA SUITE B ENCRYPTION STANDARDS

by

Jacob Paul Venema
Jonathan Lee Yee Shu

June 2009

Thesis Co-advisors:	John D. Fulp
	Richard Riehle
Second Reader:	Karl Pfeifer

Approved for public release; distribution unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 2009	Masters Thesis

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Investigation into the Impacts of Migration to Emergent NSA Suite B Encryption Standards	
6. AUTHOR(S) Jacob Venema and Jonathan Shu	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT	12b. DISTRIBUTION CODE
Approved for public release; distribution is unlimited	A

13. ABSTRACT (maximum 200 words)

As information sharing becomes increasingly necessary for mission accomplishment within the Department of Defense, the rules for protecting information have tightened. The sustained and rapid advancement of information technology in the 21st century dictates the adoption of a flexible and adaptable cryptographic strategy for protecting national security information. RSA techniques, while formidable, have begun to present vulnerabilities to the raw computing power that is commercially available today.

This thesis is a comprehensive characterization of the current state of the art in DoD encryption standards. It will emphasize the mathematical algorithms that facilitate legacy encryption and its proposed NSA Suite B replacements. We will look at how the new technology addresses the latest threats and vulnerabilities that legacy methods do not fully mitigate. It will then summarize the findings of the security capabilities of NSA Suite B standards as compared to the costs in manpower and money to implement them, and suggest how to best utilize NSA Suite B technology for the purpose of providing confidentiality, integrity and availability in an environment with real world threats.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Elliptical Curve Cryptography, ECC, Rivest Shamir and Adleman, RSA, NSA Suite B, Encryption, Digital Signature, Key Agreement, ECC Migration, Risk Mitigation	117
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

INVESTIGATION INTO THE IMPACTS OF MIGRATION FROM CURRENT DOD ENCRYPTION STANDARDS TO EMERGENT NSA SUITE B STANDARDS

Jacob P. Venema
Captain, United States Marine Corps
B.S., United States Naval Academy, 2000

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN SYSTEMS TECHNOLOGY
(COMMAND, CONTROL & COMMUNICATIONS (C3))**

Jonathan Shu
DoD Civilian
B.S., University of California, Berkeley, 2002

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SOFTWARE ENGINEERING

from the

**NAVAL POSTGRADUATE SCHOOL
June 2009**

Authors: Jacob P. Venema

Jonathan Shu

Approved by: John D. Fulp, Co-Advisor
Richard Riehle, Co-Advisor

Karl Pfeiffer
Second Reader

Peter Denning
Chairman, Computer Science Department

Dan Boger
Chairman, Information Sciences Department

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

As information sharing becomes increasingly necessary for mission accomplishment within the Department of Defense, the rules for protecting information have tightened. The sustained and rapid advancement of information technology in the 21st century dictates the adoption of a flexible and adaptable cryptographic strategy for protecting national security information. RSA techniques, while formidable, have begun to present vulnerabilities to the raw computing power that is commercially available today.

This thesis is a comprehensive characterization of the current state of the art in DoD encryption standards. It will emphasize the mathematical algorithms that facilitate legacy encryption and its proposed NSA Suite B replacements. We will look at how the new technology addresses the latest threats and vulnerabilities that legacy methods do not fully mitigate. It will then summarize the findings of the security capabilities of NSA Suite B standards as compared to the costs in manpower and money to implement them, and suggest how to best utilize NSA Suite B technology for the purpose of providing confidentiality, integrity and availability in an environment with real world threats.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

For her unwavering support, love and understanding during the writing of this thesis, Jake Venema would like to thank Karen. You have made my experience more worthwhile than any degree I could ever achieve here. To Joycey and Cecily: Thank you for your love, patience and limitless spirit these last two years. You are my light, my love, my everything. I couldn't have done it without you. I love you all.

Jonathan Shu would like to thank his employer, Defense Manpower Data Center (DMDC), for the opportunity to continue his education. Also, Jonathan Shu wishes to thank his wife, Christine Kyauk, and his daughter, Hannah Shu, for their patience, sacrifice, perseverance, and most of all for their unconditional love. It was only through their support and personal sacrifice that this thesis became reality. Hannah, your smile and kisses keep me going. I owe you big on reading your favorite Winnie the Pooh and Dora books. Thank you for your patience and your hugs.

Finally, both authors wish to express their gratitude to all those who contributed to the editing, reading and processing of this document; especially J.D. Fulp, Richard Riehle, Karl Pfeiffer, Janice Rivera, and Janis Higginbotham Your time and especially patience is most appreciated.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. MOTIVATION FOR THESIS

In August 2007, the National Institute of Standards and Technology (NIST) issued the SP 800-78-1 publication on Information Security. SP 800-78-1 specifies the timeline for mandatory Public Key Infrastructure (PKI) cryptographic key and hash migration for the next several years. This standard stipulates the implementation deadlines for several Personal Identification Verification (PIV) related cryptographic changes such as new algorithms and larger key and hash sizes. Each cryptographic change in Common Access Card (CAC) and Certificate Authority (CA) key size, key algorithms, and hash size has the potential to push issuance and usage times beyond acceptable limits for the average Department of Defense (DoD) user.

This thesis will determine some of the potential impact of this migration on CAC usage. It will describe findings obtained through lab testing of CAC cards using old and new encryption techniques and will assess some of the risk associated with a DoD-wide migration to RSA 2048 with SHA-1 keys and eventually to NSA Suite B. Migration risk will be analyzed and quantified using a critical path analysis technique.

The motivation for this research is to help identify to the federal community some prudent testing and some potentially important milestones for each of the proposed cryptographic changes. In order to do this we will focus on issues of performance and risk management.

B. CURRENT STATE OF DOD CAC METHODS AND PROCEDURES

The Department of Defense has approximately four million CAC cards in active circulation. Use of these cards is mandatory for PKI, DoD-wide interoperable access to systems, and physical access to government intranet sites. The degrees of freedom in this PKI system are very low, and any disruptions are likely to have significant and potentially long lasting repercussions. Hence, a change in any part of the CAC system must be tested carefully and meticulously prior to entering final production. SP 800-78-1 specifies a number of cryptographic enhancements and the time line for their implementation. These include a substantial increase in RSA key size from 1024 to 2048 bits, transition from SHA 1 to SHA 256, continuation of the use of the AES symmetric algorithm and eventually transition to components of NSA Suite B.

C. SCOPE

The CAC testing procedures assume proper functionality of all tested algorithms and so focus instead on performance, although observation of both aspects will occur in this thesis. Testing is divided into two parts. Part one is a comparison of 1024- and 2048-bit RSA key generation and Secure Sockets Layer (SSL) authentication. Part two compares the same metrics for RSA 1024 as they match up against ECC 256.

The recommended testing is not meant to be exhaustive but rather to indicate possible performance constraints. The DoD PIV End Point CAC performs three RSA key generations inside of the smart card for total security

assurance. This paper includes detailed statistical information on the difference between the 1024-bit keys and 2048-bit keys.

D. ENCRYPTION TAXONOMY

The modern field of cryptography includes authentication, integrity, confidentiality and non-repudiation of information. Some older cryptographic methods rely on the secrecy of their encryption algorithms. Most modern algorithms base their security on the security of keys rather than the secrecy of the method. Most modern techniques fall into one of two types: symmetric and asymmetric.

Symmetric key cryptography uses a single key for both encryption and decryption. With symmetric key cryptography, the key must be known to both the sender and the receiver. The most significant challenge to this method then becomes the distribution and management of these keys. If there are 100 people who need to communicate with one another, each one of them needs to share a common secret key with the other 99. The implication of this is that all 100 people have to keep all keys safe, which creates a challenging security situation.

Asymmetric key cryptography, or public key cryptography, came about in part to address the key management issues created by use of symmetric key cryptography. Instead of a single, secret key for each pair of users, asymmetric key cryptography requires a private and a public key for each participating user. The public key, which is used for the encryption of the message, is freely distributable. The private key is used

only to decrypt the message. Since the private key cannot be computed based solely on the knowledge of the public key, the system remains secure even though the public key is freely available. In this scenario, every user needs only keep his or her private key confidential. An added benefit of this method is that it facilitates a method for generating digital signatures for authentication and non-repudiation.

Symmetric algorithms are generally much faster to execute than asymmetric. However, asymmetric algorithms are roughly 100 to 1,000 times more difficult to break depending on the algorithm (De Clercq, 2006). In broad security practice, symmetric and asymmetric algorithms are used together so that an asymmetric key algorithm can be used to exchange a randomly generated symmetric key. The generated symmetric key can then be used to encrypt the actual message using a symmetric algorithm. Following from this idea, asymmetric ciphers are typically used for data authentication through digital signatures, for the distribution of a symmetric bulk encryption key, for non-repudiation services, and for key agreement. Symmetric ciphers support the secure exchange of information synergistically with asymmetric algorithms by bulk encrypting the actual data.

II. RIVEST SHAMIR ADLEMAN (RSA) ALGORITHM

A. INTRODUCTION

1. History of RSA Algorithm Based Encryption

The RSA algorithm, initially published in the paper "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems" in 1977 is the product of the combined efforts of Ron Rivest, Adi Shamir and Len Adleman. Named after the first initials of the MIT researchers last names, the RSA algorithm can be used for both public key encryption and digital signatures. Its security is based on the difficulty of factoring large prime integers. RSAs breakthrough became widely publicized by Martin Gardner in August 1977, in his column "Mathematical Games" in *Scientific American* magazine. At the time, the authors offered to send their full report to anyone who sent them a self-addressed stamped envelope. In spite of attempts by the NSA to stop the international distribution of the RSA source code it continued due to lack of a legal basis for the NSAs request. A more detailed version was subsequently published in the February 1978 edition of *The Communications of the ACM* thereby rendering all protests moot. Regardless, the legal battle with the U.S. Government over the RSA algorithm went on for several years. Finally, in 1982, Rivest, Shamir and Adleman formed the company "RSA" to market their Public Key Cryptography (PKC) algorithm as an electronic security product. They obtained a patent on the RSA algorithm in the U.S. only. They could not obtain an international patent because they had already published their ideas globally and most

countries bar retroactive patenting of open source concepts. In September 2000, the U.S. patent for the RSA algorithm expired, enabling software developers everywhere to freely include this PKC standard in their products (Stewart, 2009).

Whether as a licensed product, (e.g., part of Pretty Good Privacy), or implemented for private use, the RSA algorithm has become the foundation of an entire generation of public key cryptography security products. It provides secure communications between parties separated by distance that may have never met. RSA provides the ideal mechanism required for private communications over distributed electronic networks, and forms the basis of almost all the security products now in use on the Internet for financial and other private communications, including most enterprise level Public Key Infrastructure (PKI) systems, like that operated by the Department of Defense.

B. UNDERLYING CONCEPTS AND TECHNOLOGY

1. Key Generation

In order to generate the necessary keys to support RSA cryptographic operations, the following algorithm is required. Two large, random prime numbers must be generated, p and q, of approximately equal size such that their product, n, is of the required bit length. The size of an RSA key refers to the bit-length of the RSA modulus. This should not be confused with the actual number of bits required to store an RSA public key, which may be slightly more (Lenstra & Verheul, p. 8). The topic of key lengths and their associated strengths will be looked at in Chapter

III of this thesis, but in general the larger the prime numbers, the stronger the generated key.

Next, compute n = pq and compute φ = (p-1)(q-1). The value of n is known as the modulus. The value of phi is the lower limit of the boundary in which the set of numbers that are co-prime to n is contained. The upper limit is n itself.

Third, choose an integer e such that $1 < e < \varphi$, and such that the greatest common divisor (gcd) of e and φ is 1. The value chosen for e is known as the public exponent or encryption exponent or just simply the exponent.

Finally, Compute the secret exponent d, $1 < d < \varphi$, such that $(e)(d) \equiv 1 \pmod{\varphi}$. In order to compute the value for d, the Extended Euclidean Algorithm must be used to calculate $d = e^{-1} \pmod{\varphi}$. The Extended Euclidean Algorithm is used in mathematics for finding the gcd of any two integers. The computed d value is known as the secret exponent or decryption exponent.

The public key now becomes (n, e) and the private key, (n, d). All the values d, p, q and φ must be kept secret. In practice, common choices for *e* are 3, 17 and 65537 ($2^{16}+1$). These are Fermat primes, sometimes referred to as F0, F2 and F4 respectively. The formula used to derive numbers from the Fermat sequence is F(x)=2^(2^x)+1). They are chosen because they make the modular exponentiation operation faster. Also, having chosen e, it is simpler to test whether gcd(e, p-1)=1 and gcd(e, q-1)=1 while generating and testing the primes in the first step. Values of p or q that fail this test can be rejected

without further consideration. Once keys have been generated, encryption and decryption becomes a relatively simple mathematical process.

2. Encryption

Once a key of appropriate length has been generated it next falls to the process of encrypting the plaintext message. In order to encrypt, the sender must do the following:

1. Obtain the recipients public key.
2. Represent the plaintext message as a positive integer m.
3. Compute the cipher-text $c = m^e \pmod{n}$.
4. Send the cipher-text c to the recipient.

3. Decryption

In order to decrypt the message, the recipient must do the following:

1. Use their private key to compute $m = c^d \pmod{n}$.
2. Extract the plaintext from the message representative m.

4. Key Management

Anyone who wishes to sign a message or decrypt an encrypted message must have a key pair. It is common to use separate key pairs for signing messages and encrypting messages. Additionally, a user could have a key pair affiliated with his or her work and a separate key pair for personal use. Other entities may also have key pairs, including electronic devices such as modems, workstations,

Web servers (Web sites) and printers, as well as organizational entities such as a corporate department, a hotel registration desk, or a university registrars office. Key pairs allow people and other entities to authenticate and encrypt messages. (RSA Security, 2000, p. 4.1.1)

A user can generate his or her own key pair, or, depending on local policy, a security officer may generate key pairs for all users. There are tradeoffs between the two approaches. In the former, the user needs some way to trust his or her copy of the key generation software, and in the latter, the user must trust the security officer and the private key must be transferred securely to the user. Typically, each node on a network would be capable of local key generation.

Once a key has been generated, the user must register his or her public key with some central administration, called a Certification Authority. The CA returns to the user a certificate attesting to the "binding" of the users public key to certain user attributes; the users unique name/identity being one typical attribute. If a security officer generates the key pair, then the security officer can request the certification of the public key on behalf of the user.

As with all keys, distribution is important to security. Key distribution must be secured against observation (unauthorized disclosure), modification (a loss of integrity) and impersonation (a loss of authenticity). If an attacker has a way to give a legitimate user an arbitrary key that will make him believe it belongs to another legitimate user, and the attacker can intercept

transmissions between two legitimate users, then they can send their own public key (which is believed to belong to a legitimate user) and intercept any real cipher-text sent. Once intercepted the encrypted message can be decrypted with the attackers own private key, a copy of the message can be saved, the message can be re-encrypted with a legitimate public key, and the new cipher-text can be sent to the intended legitimate recipient. In principle, neither legitimate user would be able to detect the activities of the attacker. Defenses against such attacks are often based on digital certificates or other components of a public key infrastructure.

C. SECURITY CONSIDERATIONS

Factoring an RSA-modulus, n, by exhaustive search amounts to trying all primes up to the product of p and n. Finding a discrete logarithm by exhaustive search requires on the order of p operations in a finite field of numbers as large as p itself. If exhaustive search were the best attack on these systems, then bit length could be relatively small with an acceptable level of security. However, there are much more efficient and creative attacks available to attackers. Such attacks can only be defeated by the use of much larger keys, which will help to maintain acceptable security. The most efficient factoring algorithm published to date is the Number Field Sieve, invented in 1988 by John Pollard. Originally, it could be used only to factor numbers of a special form, such as the ninth Fermat number 2512 + 1. This original version is currently referred to as the Special Number Field Sieve (SNFS), as opposed to the General Number Field Sieve

(GNFS), which can handle numbers of arbitrary form, including RSA moduli. Heuristically the GNFS can be expected to require time proportional to

$$e^{(1.9229+o(1))\ \ln(n)1/3\ln(\ln(n))2/3}$$

to factor an RSA modulus n, where the o(1) term goes to zero as n goes to infinity. (Lenstra & Verheul, p.9-10)

1. RSA Integer Factorization

As previously stated, the RSA factoring problem is the task of taking eth roots modulo a composite n: recovering a value m such that $c = m^e \bmod n$, where (n,e) is an RSA public key and c is an RSA encrypted message. The simplest and most direct approach to solving the RSA problem is to factor the modulus. With the ability to recover its prime factors, an attacker can compute the secret exponent d from a public key (n,e), then decrypt c using the standard procedure. To accomplish this, an attacker factors n into p and q, and computes (p – 1)(q – 1), which then allows d to be determined from *e*. No set number of computational steps for factoring large integers on a classical computer has yet been found, however the absence of evidence is not definitive evidence of absence. It has not been proven that no polynomial-time method exists to solve the algorithm, outright.

As of 2008, the largest number factored by a general-purpose factoring algorithm was 663 bits long (RSA-200), using a state-of-the-art distributed implementation. The next largest number is probably going to be a 768-bit modulus according to Peter Montgomery in his October 2008

publication on Preliminary Design of Post-Sieving Processing for RSA-768. (Montgomery, 2008)

NIST recommends RSA keys to be at least 1,024 bits with a near term increase to 2048 bits long. With increased computing power now commercially available, 1,024-bit keys may become breakable in the foreseeable future, and will be considered insufficiently secure in the year 2010.

As previously stated, the strength of RSA encryption is based on key size. The bigger the key, the better security it provides. If the key length, n, is 300 or fewer bits, it can be factored in a few hours on a personal computer with average processing capability, using software already freely available. In 1999, keys of 512-bit length were shown to be breakable when RSA-155 was factored by using several hundred computers. Since that time, 512-bit length keys have been demonstrated to be vulnerable to factoring using commonly available hardware in as little time as a few weeks. (Fivemack, 2007)

2. Timing Attacks

In 1995, President and Chief Scientist of Cryptography Research Inc., Paul Kocher, described a new way to attack the RSA algorithm. If an attacker knows the legitimate users hardware in sufficient detail and is able to measure the decryption times for several known cipher-texts, he can deduce the decryption key quickly. This attack can be applied against the RSA signature scheme as well. In 2003, a more practical attack capable of recovering RSA factorizations over a network connection (e.g., from a SSL-enabled Web server) was found. This attack took advantage

of a weakness in the Chinese Remainder Theorem (CRT) optimization used by many RSA implementations.

One way to defeat timing attacks is to ensure that the decryption operation takes a constant amount of time for every key. However, this approach can significantly reduce performance. Instead, most RSA implementations use an alternate technique known as cryptographic blinding. Blinding makes use of the multiplicative property of RSA. Instead of computing $c^d \bmod n$, the legitimate user first chooses a secret random value r and computes $(r^e c)^d \bmod n$. The result of this computation is $rm \pmod n$ and so the effect of *r* can be removed by multiplying by its inverse. A new value of *r* is chosen for each cipher-text. With blinding applied, the decryption time is no longer correlated to the value of the input cipher-text and so the timing attack fails (Kocher, n.d.).

3. Chosen Cipher-text Attacks

A chosen cipher-text attack (CCA) is an attack model in which the cryptanalyst gathers information by choosing a cipher-text and decrypting it without previously knowing the key.

A number of seemingly secure schemes can be defeated by chosen cipher-text attacks. Early versions of RSA padding (padding will be addressed in more depth later in this chapter) used in the SSL protocol were vulnerable to a particular adaptive chosen cipher-text attack, which revealed SSL session keys. Due to flaws within the padding scheme, a practical attack against RSA implementations of the SSL protocol was found. As a result, cryptographers now recommend the use of provably secure padding schemes

such as Optimal Asymmetric Encryption Padding. Additionally, RSA Laboratories has released new versions that are not vulnerable to chosen cipher-text attacks. (Cramer & Shoup, 1998)

CCAs have implications for designers of tamper-resistant cryptographic smart cards as well. They must be particularly cognizant of the CCA threat, as these cards could conceivably fall into the hands of an unauthorized user, who might then issue a large number of chosen cipher-texts in an attempt to recover the hidden secret key.

When a cryptosystem is vulnerable to chosen cipher-text attacks, its implementers must be careful to avoid situations in which an attacker might be able to decrypt chosen cipher-texts. Specifically, the explicit requirement of a message integrity checker and the use of some form of message compression will offer protection against chosen cipher-text attacks. (Jallad, Katz & Schneier, 2003, p.12)

Chosen cipher-text attacks may be adaptive or non-adaptive. In a non-adaptive attack, the attacker chooses the cipher-text to be decrypted in advance, and does not use the resulting plain-text message to influence their next cipher-text target. In an adaptively chosen cipher-text attack, the attacker makes their target message choice adaptively, that is, the message to be decrypted is chosen based on the results of all prior decryptions. (Cramer & Shoup, 1998)

4. Branch Prediction Analysis Attacks

Branch prediction analysis, also called BPA, uses a branch predictor to determine whether a conditional branch in the instruction flow of a program is likely to be taken. Typical branch prediction analysis attacks use a spy process to statistically discover a private key when it is used to encrypt data. A more refined form of BPS known as Simple Branch Prediction Analysis (SBPA) claims to improve BPA in a way that is less calculation intensive but far more insidious and efficient. (Aciicmez, Koc & Seiffert, 2007)

While BPA attacks resemble timing attacks, where an attacker uses many execution-time measurements under the same key in order to statistically amplify some small but key dependent timing differences, SBPA dramatically improves upon standard BPA results. Using a spy process that runs simultaneously with an RSA-process, collection of almost all the secret key bits is possible during an RSA signing execution. Using SBPA 508 out of 512 bits of an RSA key were correctly identified in as few as 10 iterations. (Aciicmez, Koc & Seiffert, 2007)

In effect, SBPA is the process of analyzing a CPUs Branch Predictor states by spying on a single computation process. This one distinction provides a sharp contrast from those attacks relying on statistical methods and requiring many computation measurements under the same key. The successful extraction of almost all secret key bits by an SBPA attack against an Open SSL RSA implementation demonstrates that the often recommended blinding techniques to protect RSA against side-channel attacks are not, in and

of themselves, enough to ensure adequate protection of sensitive information. (Aciicmez, Koc & Seiffert, 2007)

5. Padding Schemes

Practical RSA implementations typically embed some form of structured, randomized padding into the value m before encrypting it. This padding ensures that m does not fall into the range of insecure plaintexts, and that a given message, once padded, will encrypt to one of a large number of different possible cipher texts.

Standards such as PKCS#1 have been designed to securely pad messages prior to RSA encryption. These standards pad the plaintext, m, with some number of additional bits, the size of the padded message will always be larger than the original message. RSA padding schemes must be carefully designed to prevent attacks, which could capitalize on a predictable message structure. At a minimum they should perform two basic tasks. The first is to add an element of randomness that can be used to convert a deterministic encryption scheme (e.g., always produces the same cipher-text for a given plaintext and key, even over separate executions of the encryption algorithm) into a probabilistic scheme. The second is to prevent partial message decryption by ensuring that an adversary is unable to recover any portion of the plaintext without being able to backwards compute the one way function. Early versions of the PKCS#1 standard used a construction that seemed to enhance RSA as a secure encryption scheme. This version was later found vulnerable to an adaptive chosen cipher-text attack. Later versions of the standard, which include Optimal Asymmetric Encryption Padding (OAEP), prevented

such attacks. The PKCS#1 standard also incorporates processing schemes designed to provide additional security for RSA signatures, for example, the Probabilistic Signature Scheme for RSA (RSA-PSS).

D. CURRENT STATE OF THE ART

1. How are RSA Encryption Techniques Being Used Today?

Since the DoD first implemented smart card technology based on CAC specifications, the Common Access Card has served as the standard ID card for millions of active duty military personnel, reservists, DoD civilian employees and contractors. The CAC is becoming the principal card used to enable physical access to buildings and controlled spaces, and is also being used to support applications such as manifesting, food service and medical and dental. It is also being used to control logical access to DoD computer networks and systems. An individuals CAC card contains their private key to be used for secure authentication to computer systems operating within a given public key infrastructure.

According to the DoDs Access Card Office, the ultimate goal of the CAC program is to create an “any card, any reader, any vendor” smart card environment.

Using RSA type encryption, CAC software combines the security of smart cards with the strength of digital certificates used for accessing networks, applications and data. These cards have enabled the DoD to migrate from passwords to digital certificates and finally to comprehensive PKI and single sign-on implementations.

RSA SecurID Passage smart card software, the platform on which CAC is established, is a standards-based smart card authentication program that ensures flexibility and information protection. It supports critical industry standards including X.509 v3 certificates, PKCS #5, #11 and #12, CAPI, SSL and qualified PC/SC readers. ("RSA Security Announces," 2001)

2. RSA Encryption and Key Management Suite

The RSA Encryption and Key Management Suite is an integrated suite of products that protect information at every layer of the OSI model while reducing complexity associated with point encryption key management techniques. Using many of the techniques discussed earlier in this chapter, RSA can minimize the risk associated with data breaches of sensitive information, intellectual property, and strategic and operational data. It can meet encryption requirements for data at rest and data in transit. It protects sensitive information stored in file systems on servers and endpoints, while also securely storing, distributing and managing encryption keys throughout their life cycle. Finally, this suite allows secure application design between elements of the DoD and private industry without incurring additional costs or further extending timelines. ("RSA Encryption and Key Management," 2009)

III. NSA SUITE B

A. INTRODUCTION

1. Background of NSA Suite B

The NSA Central Security Service states that, "the sustained and rapid advance of information technology in the 21st century dictates the adoption of a flexible and adaptable cryptographic strategy for protecting national security information." The NSA announced Suite B at the 2005 RSA Conference as a response to this requirement. Suite B will complement the existing policy for the use of the Advanced Encryption Standard (AES) that protects national security systems and information. Suite B includes cryptographic algorithms for hashing, digital signatures, and key exchange.

NSA Suite B is a subset of the cryptographic algorithms approved by the National Institute of Standards and Technology (NIST) and as such is suitable for use throughout DoD and all other governmental agencies. The entire suite of cryptographic algorithms is intended to protect both classified and unclassified national security systems and information. Beyond just the governmental applications, NSA Suite B will also provide industry with a common set of cryptographic algorithms that they can use to create products that meet the procurement needs of the U.S. Government. (National Security Agency Central Security Service, 2009)

When analyzing Suite B it is important to distinguish what it is and what it is not. Suite B only specifies the cryptographic algorithms to be used. There are many other

competing factors that determine whether a particular device that implements a set of cryptographic algorithms should be used to satisfy a given security requirement. The quality of the implementation of the cryptographic algorithm in software, firmware, or hardware must be sufficient. Operational requirements associated with DoD approved key and key management activities must be commensurate. Another point for consideration is the sensitivity or other access restrictions of the information to be protected (e.g., SECRET, TOP SECRET, NOFORN, FOUO, etc.). Finally, the operational requirements for interagency and international interoperability will always be a factor. The processes by which these factors are addressed are outside the scope of cryptographic technology. The primary focus of Suite B is simply protecting the information. (National Security Agency Central Security Service, 2009)

B. UNDERLYING CONCEPTS AND TECHNOLOGY

1. Elliptic Curve Cryptography (ECC)

Over the past three decades Internet communications have been secured by the earliest generations of public key cryptographic algorithms, most of which were developed in the middle to late 1970s. They have formed the basis for key management and authentication, Web traffic and secure e-mail. These public key techniques revolutionized cryptography as it had been used and understood; however, newer techniques have been developed that offer better performance and increased security. Specifically, Elliptic Curve (EC) techniques, which were independently created by

Neal Koblitz and Victor Miller, offer an exciting way ahead for the science of cryptography.

In a recent posting on the NSA website, they state that, "the best assured group of new public key techniques is built on the arithmetic of elliptic curves. While currently elliptic curves do not offer many noticeably significant benefits over existing public key algorithms, over time the evolving threat posed by eavesdroppers and hackers with access to greater computing resources will demonstrate what elliptic curves can offer." In general, current PKI methods respond to new attacks by relying on, and when necessary dramatically increasing, their key sizes. Elliptic Curve Cryptography (ECC) allows for better security with a significant reduction in key size. Moreover, "ECC has to date exhibited no vulnerabilities to increasingly strong attack algorithms, but has remained secure with little required adaptation" (National Security Agency Central Security Service, 2009).

Presently, the methods for computing elliptical curve mathmematics are much less efficient than those for factoring or computing integer factorization schemes (such as the ones used in RSA encryption). As a result, shorter key sizes can be used to achieve comparable security to conventional public-key cryptosystems. (Lenstra & Verheul, 2000) This, in turn, has the potential to lead to less memory requirement and improved performance vis-à-vis smaller keys and faster computations. These advantages are especially important in environments where processing power, storage space, bandwidth, or power consumption is constrained.

a. ECC Key Generation

A generic Elliptic Curve (EC) system consists of a public key of a finite field GF(p) of size p, a generator g of the multiplicative group GF(p), and an element y of GF(p) that is not equal to 1. In an EC system, g generates a subgroup, q, of the group of points on an elliptic curve, E, over the finite field GF(p). The security is based on the difficulty of computing discrete logarithms in the subgroup generated by g. These subgroup logarithms can be computed only if all the discrete logarithms in the full group of points on an elliptic curve over a finite field can be computed. This computational procedure is known as the Elliptic Curve Discrete Logarithm (ECDL) problem. There is currently no better method to solve the ECDL problem other than to solve the problem in all cyclic subgroups and then combine the results. The difficulty of the ECDL problem depends on the size of the largest prime divisor of the order of the group of points of the curve, which is close to p. For this reason, p, E, and q are usually chosen such that the sizes of p and q are close. The security of EC systems, then, relies on the size of q. The size of an EC key refers to the bit-length of the subgroup size q. The actual number of bits required to store an EC public key could conceivably be substantially larger than the EC key size q, since the public key contains p, E, g, and y as well. (Lenstra & Verheul, 2000)

ECC systems use two kinds of curves. The first kind, Pseudo-random curves, are those whose coefficients are generated from the output of a seeded cryptographic hash. If the seed value is given along with the

coefficients, it can be verified that the coefficients were generated by that method. The second kind, Special curves, are those whose coefficients and underlying field have been specifically selected to optimize the efficiency of the elliptic curve operations. (U.S. Dept of Commerce, 2000, p. 29)

b. Elliptic Curve Digital Signature Algorithm

Fundamentally a digital signature is simply the digital version of a handwritten signature that is used every day to demonstrate that whatever was signed was done by one and only one person. A digital signature is represented in a computer as a string of binary digits that are dependent on the signers private key, as well as the contents of the message. A digital signature is computed using a set of rules and a set of parameters such that the identity of the signatory and integrity of the data can be verified. An algorithm provides the capability to generate and verify the signatures. Signature generation makes use of a private key to encrypt a hash of the data being signed. An adversary, who does not know the private key of the signatory, cannot generate the correct signature of the signatory. More plainly stated; signatures cannot be forged. Signature verification makes use of a public key, which corresponds to, but is not the same as, the private key. By using the signatorys public key, anyone can verify a correctly signed message. A means of associating public and private key pairs to the corresponding users is required. That is, there must be a binding of a users identity and the users public key. This binding may be certified by a mutually trusted and unbiased third party.

Possible disputes that could arise will do so either when a signer tries to deny a signature they legitimately created, or when a forger makes a fraudulent claim. For both signature generation and verification, the data, which is referred to as a message, is reduced by means of the Secure Hash Algorithm (SHA-1) as specified in the Federal Information Processing Standards (FIPS) Publication 180-2.

Digital signatures can be used to provide basic cryptographic services such as data integrity, which assures that data has not been altered by unauthorized means, data authentication that assures that the source of data is as claimed and finally, non-repudiation, which assures that an entity cannot deny previously sent transmissions. (U.S. Department of Commerce, 2000)

The Elliptic Curve Digital Signature Algorithm (ECDSA) is the elliptic curve flavor of the Digital Signature Algorithm used in legacy cryptosystems. ECDSA was first proposed in 1992 by Scott Vanstone in response to the NISTs request for public comments on their first proposal for a Digital Signature Standard (Vanstone, 1992). It was accepted in 1998 as an ISO (International Standards Organization) standard (ISO 14888-3), and subsequently in 1999 as an ANSI (American National Standards Institute) standard (ANSI X9.62). Finally, in 2000 it was accepted as an IEEE (Institute of Electrical and Electronics Engineers) (IEEE 1363-2000) and a FIPS standard (FIPS 186-2). (Johnson & Menezes & Vanstone, 2001)

An ECDSA key pair (public and private) is associated with a particular elliptic curve. The public key is a random multiple of the base point, while the

private key is the integer used to generate the multiple. In other words, given two points *G* and *Y* on an elliptic curve such that *Y = kG*, k is the private key and Y is the public key. Finding k is another example of the ECDL problem.

A large part of the digital signature process involves legitimizing the key pairs used. Proving ownership and validity greatly increases the quality of the assurance of the signature. Public key validation ensures that a public key has the requisite mathematical properties. Successful execution of the validation process demonstrates that an associated private key logically exists, although it does not demonstrate that someone actually has computed the private key. More importantly, it does not demonstrate that the claimed owner actually possesses the private key. Practical reasons for performing public key validation include both prevention of malicious insertion of an invalid public key and detection of inadvertent coding or transmission errors. (Johnson & Menezes & Vanstone, 2001)

In order to prevent an entity from claiming a fraudulent public key, the CA should require all entities to prove possession of the private keys corresponding to its public keys before the CA certifies the public key. This proof of possession can be accomplished in a variety of ways. One such method is to require all entities to sign a message of the CAs choice. The other is by using "zero-knowledge" techniques. It is noteworthy to highlight the fact that proof of possession of a private key provides different assurances from public key validation. The former demonstrates possession of a private key even though it may

correspond to an invalid public key, while the latter demonstrates validity of a public key but not ownership of the corresponding private key. Doing both provides a higher level of assurance regarding digital signatures.

c. Elliptic Curve Diffie-Hellman Key Agreement

The elliptic curve Diffie-Hellman scheme is a key agreement scheme based on ECC. It is designed to provide either unilateral or mutual key authentication, known-key security, and forward secrecy. It does this under the assumption that issues involving authenticated public key exchange and key pairs' states (ephemeral vs. static) have been resolved in accordance with NIST recommendations (Certicom Research, 2000).

Secret cryptographic keying material may be electronically established between parties by using either a key agreement scheme or a key transport scheme. During key agreement both parties contribute to the shared secret and by extension the derived secret keying material. The secret keying material to be established is therefore never sent directly to one person or the other. Instead, information is exchanged between both parties that then allows each to derive the secret keying material.

An alternative option to the key agreement method described above is the use of an asymmetric key based key transport scheme. During key transport one party selects the secret keying material. The encrypted or "wrapped" secret keying material is transported from the sender to the receiver. (Barker & Johnson & Smid, 2007)

The elliptic curve Diffie-Hellman technique examined here generates a field element from a secret key owned by one entity and a public key owned by a second entity in such a way that when both execute the key exchange technique with corresponding keys as input, they will compute the same field element. The primary security requirement is that an attacker who sees only one entity or the others public key should be unable to compute the shared field element.

The requirement that an attacker be unable to derive the shared field element is a direct result of the requirement that the elliptic curve Diffie-Hellman problem (ECDHP) be sufficiently difficult to solve. The ECDHP is closely related to the ECDLP mentioned earlier. If the ECDLP is easy then the ECDHP is also. The converse is also true (Boneh & Lipton, 1996). Many key agreement schemes based on Diffie-Hellman actually rely on the stronger requirement that the shared field element is not just sufficiently difficult for an attacker to predict, but that the element actually looks random to the attacker. (Certicom Research, 2000)

For key exchange, NSA Suite B calls for the use of ECDH. According to the NSA, ECDH is appropriate for incorporation of Suite B into many existing Internet protocols such as the Internet Key Exchange (IKE), Transport Layer Security (TLS), and Secure MIME (S/MIME).

d. Secure Hash Algorithms

There are currently available four secure hash algorithms, SHA 1, SHA 256, SHA 384, and SHA 512. All four of the algorithms are iterative, one-way hash functions

that can process a message to produce a condensed representation called a message digest. The result is that these algorithms can be used to verify a messages integrity. Any change to the original message will result in a different message digest. This capability is useful in the generation and verification of digital signatures and message authentication codes. Every secure hash algorithm can be described by its two stages of preprocessing and hash computation. (U.S. Department of Commerce, 2002)

Preprocessing involves padding a message, parsing the padded message into m-bit blocks, and setting initialization values for the eventual hash computation. The hash computation generates a message schedule from the padded message and uses that schedule, along with functions, constants, and word operations to iteratively generate a series of hash values. The final hash value generated by the hash computation is used to determine the message digest. The four algorithms differ most significantly in the number of "bits of security" that are provided for the data being hashed, which is directly related to the message digest length and to the deviation of the algorithms output distribution (i.e., how uniformly likely are all possible output strings). When a secure hash algorithm is used in conjunction with another algorithm, there may be additional requirements that mandate the use of a secure hash algorithm with a minimum number of bits of security. For example, if a message is being signed with a DSA that provides 128 bits of security, then that signature algorithm may require the use of a

secure hash algorithm that also provides 128 bits of security (e.g., SHA-256). (U.S. Department of Commerce, 2002)

Finally, the four algorithms differ in terms of the size of the blocks and words of data that are used during hashing. Table 1 below lists the basic properties of all four secure hash algorithms.

Algorithm	Message Size (bits)	Block Size (bits)	Word Size (bits)	Message Digest Size (bits)	Security[2] (bits)
SHA-1	$< 2^{64}$	512	32	160	80
SHA-256	$< 2^{64}$	512	32	256	128
SHA-384	$< 2^{128}$	1024	64	384	192
SHA-512	$< 2^{128}$	1024	64	512	256

Table 1. Secure Hash Algorithm Properties [From U.S. Department of Commerce/National Institute of Standards and Technology (2002)]

C. SECURITY CONSIDERATIONS

1. Message Encryption Criteria

The majority of public key systems in use today use 1,024-bit key parameters. NIST has recommended that these 1,024-bit key lengths be upgraded to something providing more security no later than 2010. After that, NIST recommends that they be upgraded once again to something providing even more security. One course of action would be to increase the key size up to the next level of 2048 bits. Another viable option is to move from first generation public key algorithms to elliptic curve algorithms.

Key bit length for a symmetric encryption algorithm is a common measure of security. Table 2 gives some key sizes recommended by NIST to protect keys used in conventional encryption algorithms like the Data Encryption Standard (DES) and the Advanced Encryption Standard (AES) together with the equivalent key sizes for RSA, Diffie-Hellman and elliptic curves.

Symmetric Key Size (bits)	RSA and Diffie-Hellman Key Size (bits)	Elliptic Curve Key Size (bits)
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

Table 2. NIST Recommended Key Size Equivalents[From National Security Agency Central Security Service (2009)]

Consistent with CNSSP-15, Elliptic Curve Public Key Cryptography using the 256-bit prime modulus elliptic curve as specified in FIPS-186-2 and SHA-256 is appropriate for protecting classified information up to the SECRET level. Use of the 384-bit prime modulus elliptic curve and SHA-384 are necessary for the protection of TOP SECRET information.

In order to use RSA or Diffie-Hellman to protect 128-bit AES keys one should use 3072-bit parameters, which are three times the size of those in use throughout the internet today. The equivalent (strength) key size for elliptic curves is only 256 bits. It rapidly becomes

evident that as symmetric key sizes increase, the required key sizes for RSA and Diffie-Hellman increase at a much faster rate than the required key sizes for elliptic curve cryptosystems in order to achieve the equivalent security strength. Hence, elliptic curve systems offer more security per bit increase in key size than either RSA or Diffie-Hellman public key systems. (National Security Agency Central Security Service, "The Case for Elliptical Curves," 2009)

2. ECC Vulnerabilities

In general, the best attacks on the elliptic curve discrete logarithm problems have been brute-force. The absence of algorithm specific attacks seems to indicate that shorter key sizes for elliptic cryptosystems appear to give similar security as much larger keys that might be used in cryptosystems based on the discrete logarithm problem or integer factorization. That stated, there are more efficient attacks that exist for certain choices of elliptic curves. According to Menezes, Okamoto, and Vanstone the elliptic curve discrete logarithm problem has been reduced to the more easily solvable traditional discrete logarithm problem for certain specific curves (Menezes, Okamoto, and Vanstone, 1990). The implication of this is that the same size keys as are used in more traditional public key systems are now required for those specific elliptical curves. However, these instances of vulnerability are readily classified and easily avoided and therefore do not pose much of a problem.

In 1997, elliptic curve cryptography began to receive more attention from researchers looking to test its

security. By the end of the nineties, there were no major improvements found to be necessary in the reliability of EC cryptosystems. According to RSA Laboratories, the longer this situation continues, the more public confidence will grow that ECC really does offer as much strength as advertised. However, there is some evidence that the use of special elliptic curves, which provide very fast implementations, might allow new specialized attacks. As a starting point, the basic brute-force attacks can be improved when attacking these curves. Continued research into elliptic curve cryptosystems might eventually create the same level of widespread trust as in other public-key techniques however, "the use of special purpose curves will most likely always be viewed with extreme skepticism." (RSA Security, 1998)

3. ECDSA Vulnerabilities

The security objective of ECDSA is to be "existentially unforgeable" against a chosen message attack. The goal of an adversary who launches such an attack against a legitimate entity is to obtain a valid signature on a single message, after having obtained the legitimate entitys signature on a collection of other messages of the adversarys choice.

According to Certicoms publication on ECDSA, "Some progress has been made in trying to prove the security of ECDSA, albeit in theoretical models. Slight variants of DSA and ECDSA (but not ECDSA itself) have been proven to be existentially unforgeable against chosen message attack…under the assumptions that the discrete logarithm problem is hard and that the hash function employed is a

random function. ECDSA itself has been proven secure by Brown under the assumption that the underlying group is a generic group and that the hash function employed is collision resistant."

All possible attacks on ECDSA can be categorized into one of three possible classifications. The first is attacks on the elliptic curve discrete logarithm problem. The second is attacks on the hash function employed. The third falls into the ubiquitous "other attacks" category. (Johnson & Menezes & Vanstone, 2001, p.28)

4. ECDH Vulnerabilities

A direct assault on the ECDH problem is not the only way an attacker might attempt to break the Diffie-Hellman key agreement scheme. ECDH key agreement schemes could also be susceptible to small subgroup attacks (Johnson, 1996)(Lim & Lee, 1997) in which an adversary substitutes a users public key with a point of small order in an attempt to coerce a different entity (or user) to calculate a predictable field element using one of the DH primitives. A successful attack of this type could result in the compromise of a session key shared by two entities, or in a worst-case scenario, even the compromise of one of the entitys secret keys.

Two defenses recommended against this attack are either to validate a users public key and use the standard Diffie-Hellman primitive, or partially validate the entitys public key and use the cofactor Diffie-Hellman primitive. Which defense is appropriate in a given situation will depend on issues like whether or not interoperability with existing use of the standard' Diffie-Hellman primitive is

desirable (the first defense interoperates while the second does not), and what the efficiency requirements of the system are (the second defense is usually more efficient). (Certicom Research, 2000)

D. CURRENT STATE OF THE ART

To date ECC adoption in the industry has been slow, but is gaining momentum thanks to the recent NSA endorsement of its advantages over RSA. There have been a number of important industries and government organizations who are adopting ECC, but most notably eGovernment IDM (Identity Management) initiatives in Austria. Austria began a phased implementation of Health insurance e-cards in May 2005 and was completely running by November of the same year. Health insurance e-cards use elliptic curve cryptography, NIST recommended 192 bit prime field curve. (Austrian Profile, 2007)

Adoption of ECC by smart card vendors has been also steadily increasing. In January 2007, one of the smart card chipmakers, Infineon, announced that Certicom Incs Suite B Power Bundle will be included in the Infineon smart card microcontrollers, which will comply with USGs Suite B specifications by providing Infineon ECC-enabled smart card microcontrollers and Certicom software tools for the USGs Personal Identity Verification (PIV) cards and other applications. (Certicom, 2007) Additionally, Oberthur Card Systems just recently received FIPS 140-2 Level 3 certification on their 128K smart card with ECC.

Another major industry that is adopting ECC is the utility industry, where they are trying to modernize their meter reading and data collection system with advanced

metering infrastructure. In May of 2008 Certicom launched a new Device Authentication Service for ZigBee Smart Energy. The service uses ECC to secure wireless data communication and authenticate smart metering devices.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. COMPARE AND CONTRAST

A. INTRODUCTION

In this chapter, we will compare and contrast the performance, advantages, disadvantages, and usability of ECC and RSA based cryptosystems.

B. DATA COLLECTION AND RESEARCH

The data collection methodology used in this thesis includes both results from our own testing as well as from findings from independent research companies like Certicom Inc. and RSA.

1. Empirical Data

As the DoD prepares to move forward with CAC platform cryptographic migration from RSA 1024 to RSA 2048, the CAC Test Lab (CTL) headed by one of the authors of this thesis, J. Shu, and located within the Defense Manpower Data Center (DMDC) has performed exhaustive benchmark testing in order to determine the average time required to produce CAC cards personalized with End Entity (EE) RSA 2048 Certificates. CAC issuance infrastructure testing of RSA 2048 keys and SHA-1 signatures are part of the requirement for the cryptographic migration test plan. The CTL issued a paper on the results of issuance testing with enlarged key sizes from EE RSA 1024 to EE RSA 2048 bit key length and SHA-1 signature done from a Real-time Automated Personnel Identification System (RAPIDS) workstation. These results are the basis for our comparative analysis of the two RSA key lengths.

2. Methods

RAPIDS is the DoDs main form of CAC issuance. As such, the DMDC CTL conducted an in-depth analysis of the RAPIDS logs, which were found to contain a wealth of information regarding key generation times. The CTL approach was to look at the Application Protocol Data Unit (APDU) commands embedded into the RAPIDS logs and in conjunction with the time stamps associated with the commands, measure the individual specific time lapses. In this way, they were able come up with a quantifiable value associated with the establishment of each key generation. To compensate for the considerable size of each log the CTL engineers developed data pattern recognition applications, which were used to examine and sift through the RAPIDS logs.

Pattern recognition applications work by systematically processing every line from a RAPIDS log. The CTL application used a recursive process to either filter out unimportant lines or to extrapolate significant data points from relevant lines. In this fashion the CTL was able to process an entire directory worth of files, so that multiple RAPIDS log files could simultaneously be examined. Once the information from the log files had been culled, the exact amount of time needed to complete each key generation was collected. It is important to note that the CTL focused on APDU information exchange between the CAC and the RAPIDS station with particular regard to the time required to complete the communication. The CTL did not include an investigation of the impact on networking; nor will this thesis.

3. Testing

Each testing station was configured to record issuance times for each card issued. In order to most effectively compare issuance and key generation time, a critical comparison of smartcard RSA 'on card' key generation using 2048-bit and 1024-bit keys was conducted.

C. RSA 1024 VS. RSA 2048

Presently the DoD CAC performs three RSA key generations inside of the smart card for total security assurance. After the CTL collected the RSA key generation data it was then able to calculate a statistical analysis on the timing difference between RSA 1024-bit and 2048-bit keys.

1. On-Card Key Generation Analysis

As previously stated, the algorithm for RSA key generation involves finding two large prime numbers from which key pairs are able to be generated. Finding the two large prime numbers is the most time consuming step in the RSA key generation process. Using a random number as a starting point to find the two large prime numbers they are then multiplied together to form the composite number. This composite is the key strength (e.g., 2048 bits or 1024 bits). The security of RSA is based on the difficulty of calculating the prime factors of large composite numbers.

The time difference between 1024 and 2048 bit "on card" RSA key generation can be substantial due to the simple fact that larger numbers are much more difficult to factor than smaller ones. The difference in finding factors of a composite number of 2048 bits as compared to a

composite number that is a factor of 1024 bits is non-trivial. In fact, although the 2048-bit key is twice the size of 1024-bit key, the 2048-bit key generation times takes about nine to ten times longer to complete. Figure 1 shows the average key generation time for both the RSA 2048 and RSA 1024-bit lengths. The average time for the 2048 bits is approximately 48 seconds and the average time for the 1024 bits is approximately 5 seconds for each key generation.

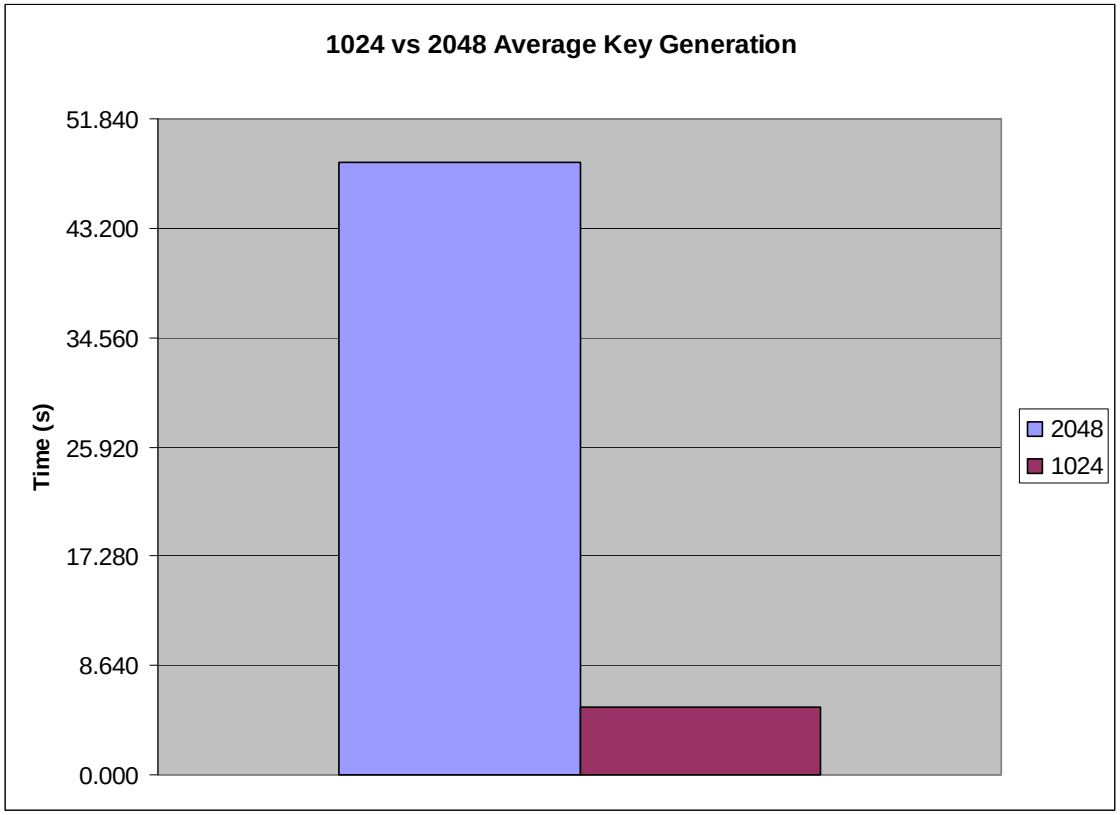

Figure 1. Average Key Generation Time

2. RSA 2048 Key Generation

Figure 2 shows the results of the total time it took to complete key generations for the 2048-bit keys. The majority of the key generations took approximately 30 to 80

seconds to be generated. The average key generation occurred around approximately 45 seconds.

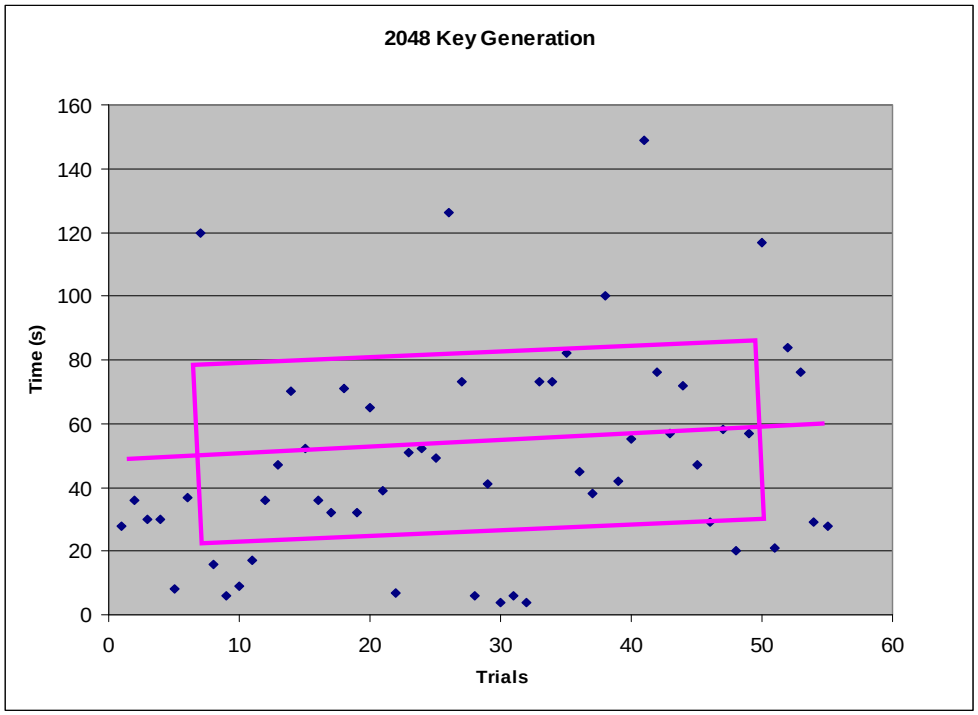

Figure 2. 2048-Bit Key Generation Times

Figure 3 also shows the results for the time it took to generate a key for the 2048 bit, however this second plot also includes a confidence interval showing where a key generation is most likely to be completed.

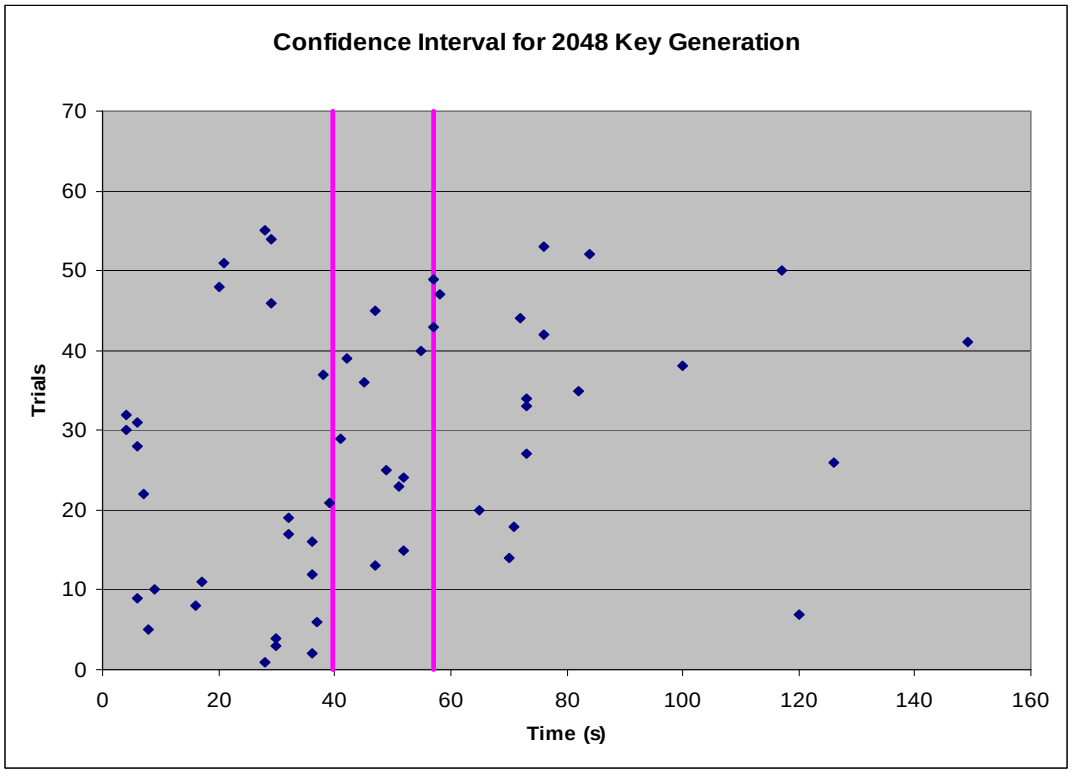

Figure 3. 2048-Bit Key Generation Times w/ CI

Figure 4 shows the probability distribution of the time it took to generate a key for the 2048 bits. It is most likely that key generations were completed in the range of approximately 30 to 75 seconds. A key is most likely to be generated at approximately 48 seconds. CTL uses the normal distribution or Gaussian distribution to describe data that clusters around a mean or average. The probability density function for a normal distribution is given by the formula

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right),$$

where μ is the mean, σ is the standard deviation (a measure of the "width" of the bell), and exp denotes the exponential function.

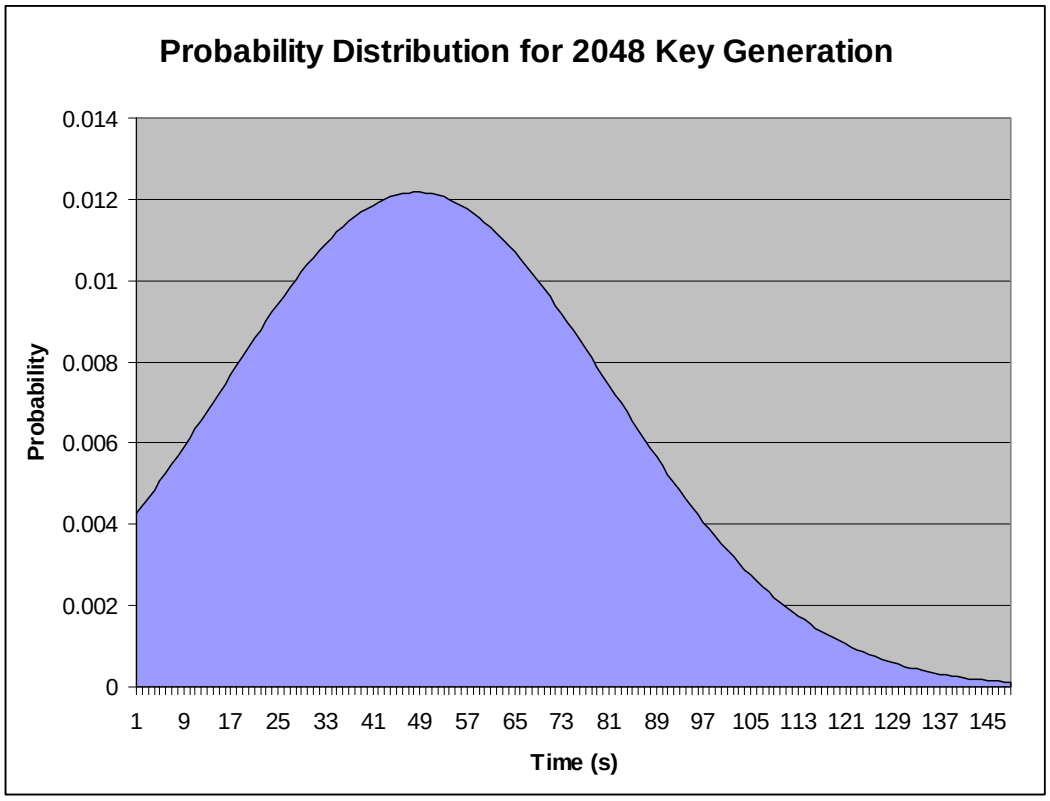

Figure 4. 2048-Bit Key Generation Probability Distribution

3. RSA 1024 Key Generation

Figure 5 shows the results of the total time it took to complete key generations for the 1024-bit keys. The majority of the key generations took approximately 2 to 7 seconds to be generated. The average key generation occurred at approximately 5.5 seconds.

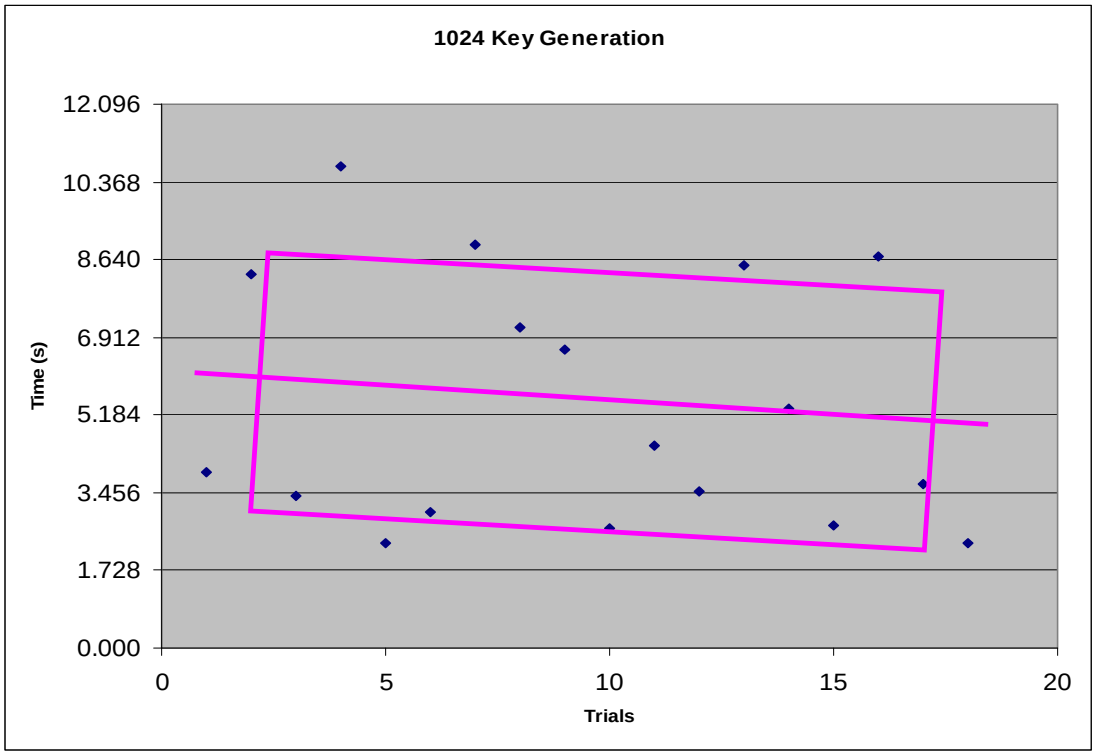

Figure 5. 1024-Bit Key Generation Times

Figure 6 also shows the results for the time it took to generate a key for the 1024-bit keys, however the plot also includes a confidence interval showing where a key generation is most likely to be completed.

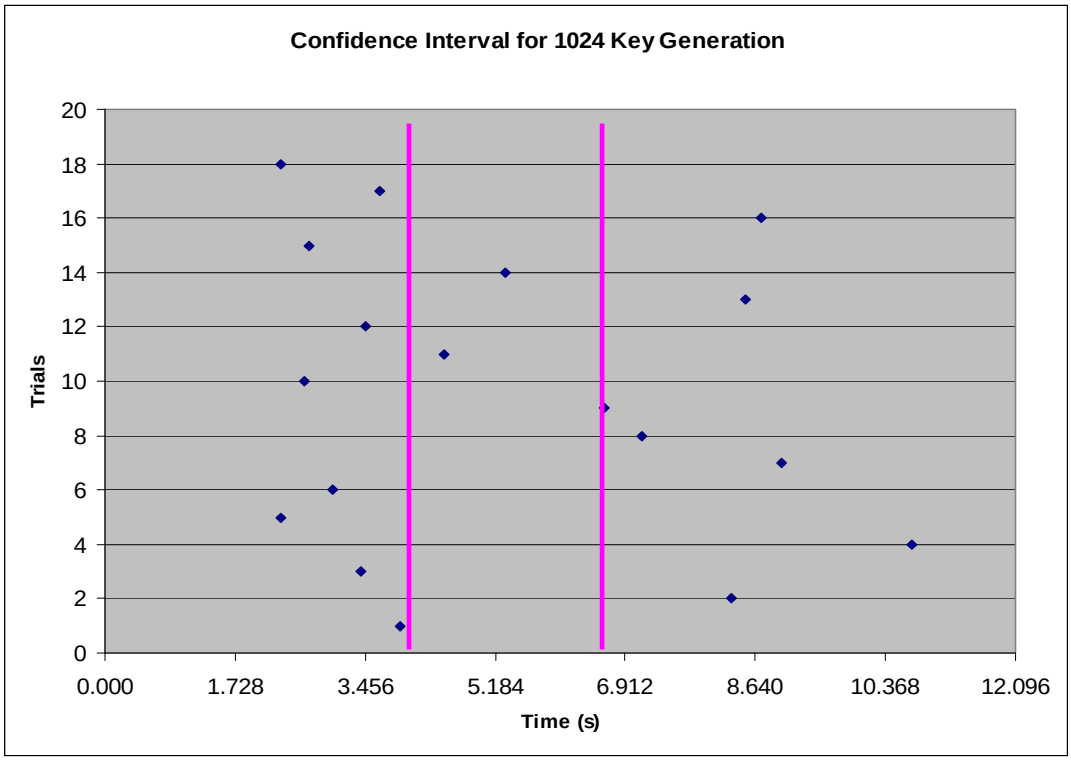

Figure 6. 1024-Bit Key Generation Times w/ CI

4. Encryption/Decryption/Signing Comparison

In an attempt to compare the two RSA key length metrics even further, results from recent DISA testing demonstrating everyday usages were studied. The following figures were published at the 2009 DoD Identity Protection Management (IPM) Conference in Miami, Florida.

Figure 7 plots a side-by-side comparison of RSA 1024 and RSA 2048 digital signatures. There are no obvious significant differences in the timing however, it is noteworthy that the time required to complete any given signature is non-trivial; in some cases taking longer than twelve seconds to sign an email with a 1 MB attachment.

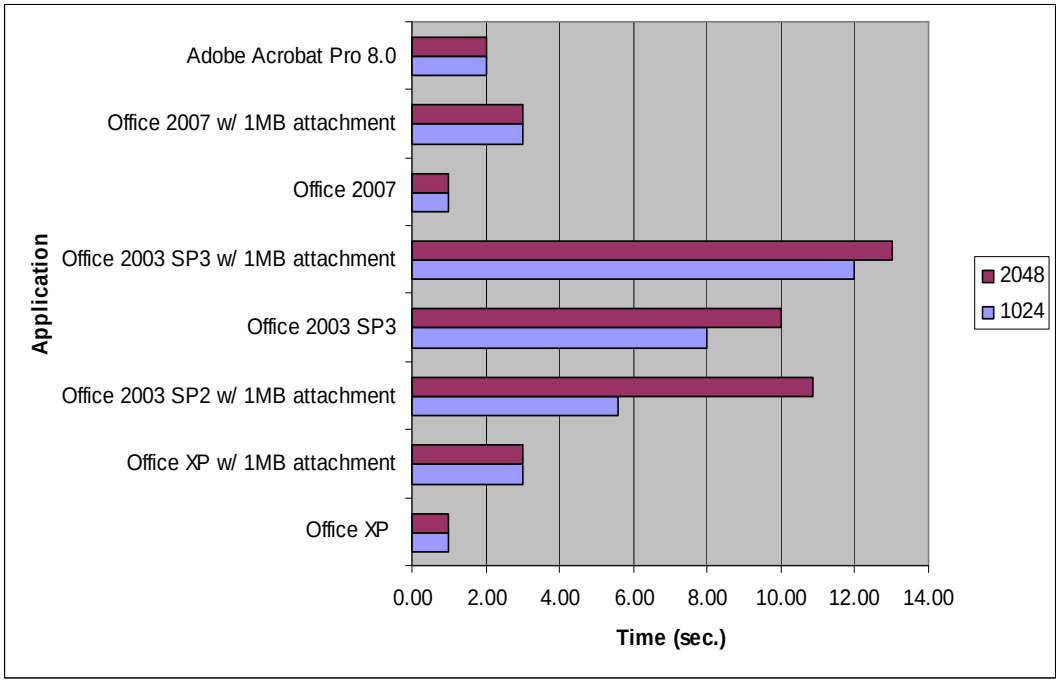

Figure 7. Digital Signature Comparison

Figure 8 compares the encryption speeds for various scenarios involving email and corresponding applications that are commonly used when transmitting data. Most encryption times are relatively comparable and so do not distinguish one key size over the other.

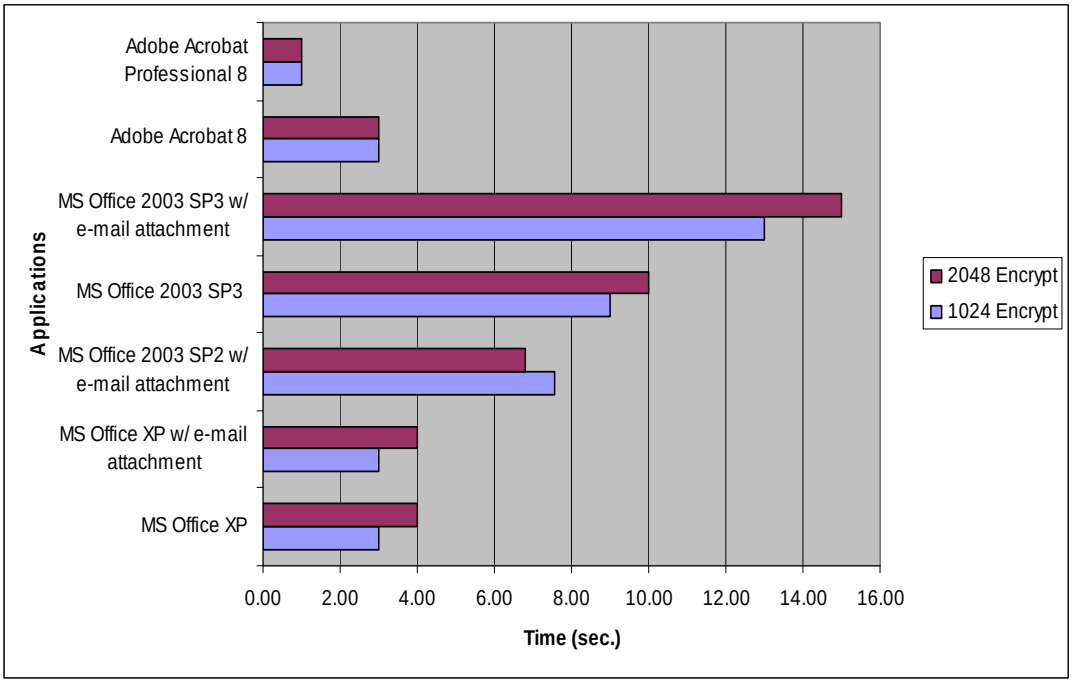

Figure 8. RSA 1024 vs. RSA 2048 Email Encryption Timing

Figure 9 below illustrates a similar comparison, in this instance focusing on the decryption times. The reader will note that the decryption times are even more difficult to distinguish since they are similar in almost every scenario save the MS Office 2003 SP2 with email attachment.

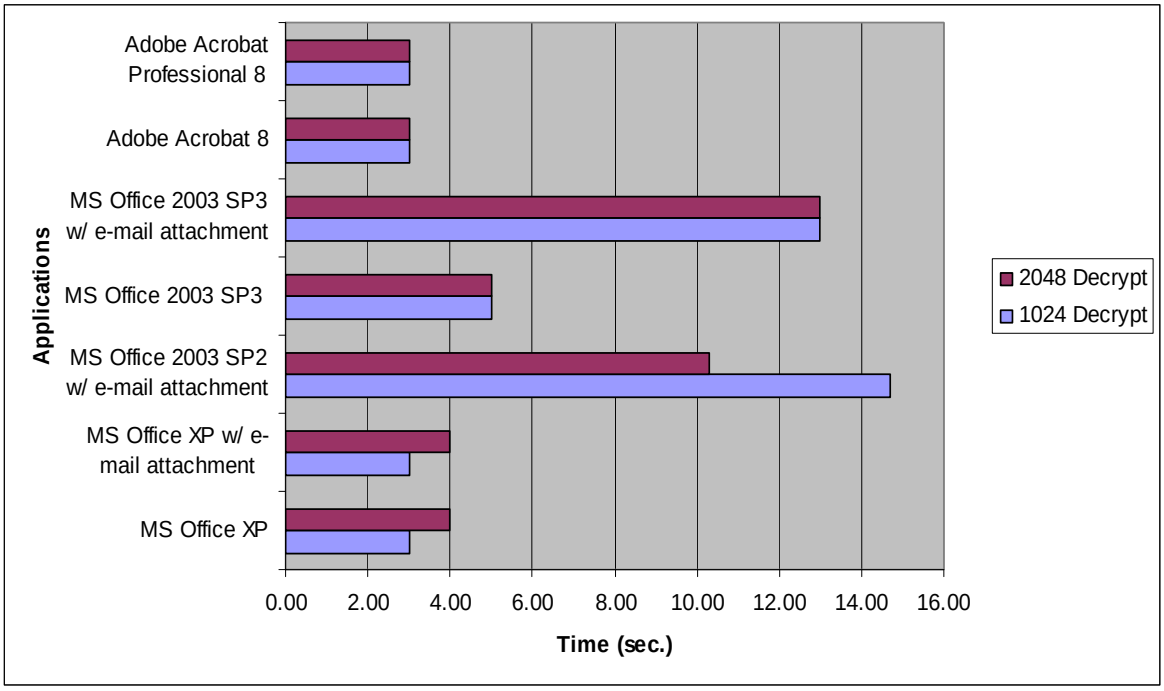

Figure 9. RSA 1024 vs. RSA 2048 Email Decryption Timing

The most likely reason for the similarities in encryption/decryption timing for the two key lengths is the fact that most of the cryptographic operations are done on the computer (not on the smart card), which means that faster resources are more freely available.

D. ECC AND RSA CRITICAL COMPARISON

1. ECC Data

At the time of the writing of this thesis, the authors did not have access to a smart card platform with ECC implementation. As such, collection of performance data similar to on-card key generation was not possible. The focus of our research was on experiments that others have done in comparing RSA and ECC, from which we have drawn conclusions.

a. Certicom Study

In May of 1998 Certicom Inc., the company that owns most ECC related patents, published a paper on ECC implementations in smart cards. Certicom Inc. has run benchmarks on a number of different platforms. Table 3 below is a sample benchmark in Solaris (167 MHz Ultra SPARC running Solaris 2.5.1) platforms to test the performance of Certicoms 163-bit ECC implementation relative to 1024-bit RSA implementations. The 1024-bit RSA key pair generation (4.7 seconds) is significantly slower than 163 bit ECC (.0038 seconds). Signing speeds are also consistently many times faster than those of RSA due to the fact that the mathematical operations used for signatures and encryption/decryption, are completely different. (Certicom, 2000)

Function	Security Builder 1.2 163-bit ECC (ms)	BSAFE 3.0 1,024-bit RSA (ms)
Key Pair Generation	3.8	4,708.3
Sign	2.1 (ECNRA)	228.4
	3.0 (ECDSA)	
Verify	9.9 (ECNRA)	12.7
	10.7 (ECDSA)	
Diffie-Hellman Key Exchange	7.3	1,654.0

Table 3. Solaris Sample Benchmark [From Certicom, 2000]

b. Research In Motion (RIM) Study

The maker of BlackBerry, Research In Motion (RIM), conducted a comparative analysis of RSA and ECC for 128-bit security strength in 2004. The similarity between the DoD CAC and a typical BlackBerry device is the use of a public key mechanism to manage the number of required keys. RIM chose the Simple Password Exponential Key Exchange

(SPEKE), which is the same as the previously discussed Diffie-Hellman key agreement protocol, except that the hash of the password is used as the generator of the group.

NIST recommends 256 bits of security for classified government communications. The RIM team tested 512-bit Elliptic Curve Diffie-Hellman (ECDH), equivalent to 15360-bit RSA or 15360-bit Diffie-Hellman according to NIST Publication 800-57. Each of these algorithms have their own advantages and trade-offs. Large keys have an impact on the performance of power and bandwidth constrained devices. RIM ran a series of tests to determine the performance levels of the above three algorithms for key generation, encryption/verification (with a public key) and decryption/ signature (with a private key).

Generally, ECC had the fastest times for a general purpose cryptosystem. RSA, however, is good for situations where only public key verification is needed. In the end, RIM decided to implement a hybrid solution for their BlackBerry device. ECC is their preferred choice of cryptosystem for systems requiring a high level of security, key generation, or private key operations, however, for operations involving only public keys (e.g., verifying a signature), RSA is RIMs preferred choice since an RSA public key operation is so much faster. RSA encryption is used on the BlackBerry for signature verification. The timing results listed in Table 4 below were taken using a BlackBerry 7230 with 128-bit security.

	ECC (256)	RSA (3072)	DH (3072)
Key generation	166 ms	Too long	38 s
Encrypt or verify	150 ms	52 ms	74 s
Decrypt or sign	168 ms	8 s	74 s

Table 4. ECC 256 vs RSA 3072 Performance Comparison [From Certicom, 2004]

c. Palm Device Study

Neil Daswani from Stanford University studied performance of various ECC and RSA cryptographic primitives on various Palm OS platforms. Figure 10 shows interesting performance data on the Palm OS. In studying the Wireless Transport Layer Security (WTLS) protocol, the cryptographic requirements of the protocol, and the time required to execute the required operations, Daswani found that 1024-bit RSA based handshakes can be up to twice as fast as 163-bit ECC based handshakes for *server*-authenticated WTLS connections, and that 163-bit ECC based handshakes are at least 8 times as fast as 1024-bit RSA-based handshakes for *mutually*-authenticated WTLS connections. (Daswani, n.d.)

	New Palm VII (Dragonball-EZ, 20MHz, PalmOS v.3.2.5) (ms)	Palm V (Dragonball-EZ, 16.6MHz, PalmOS v.3.3) (ms)	Old Palm VII (Dragonball, 16.6MHz, PalmOS v. 3.1) (ms)
ECC Benchmarks (163-bit)			
Key Generation	372.4	514	556
Key Expansion[1]	254.8	350	378
Diffie-Hellman Key Agreement	335.6	462	500
ECC-DSA Signature Generation	514.8	713	773
ECC-DSA Signature Verification	1254	1740	1885
RSA Benchmarks(1024-bit)[2]			
Private Encrypt	21734	27808	29628
Public Decrypt (e=3)	598	758	790
Public Decrypt (e=65537)	1482	1860	1966
Public Encrypt	622	798	834

Figure 10. Execution times for RSA and ECC cryptographic primitives [From Daswani, n.d.)

d. Trusted Platform Module (TPM) Study

In 2007 Beijing University of Technology conducted a study on ECC implementation of the Trusted Platform Module. The TPM is a microcontroller that stores keys, passwords and digital certificates. It typically is affixed to the motherboard of a PC, but could potentially be used in any computing device that requires these functions. The nature of the TPM is similar to smart cards in that they are both hardware based protection solutions and they both aim to be bandwidth and power efficient devices. The TPM ensures that the information stored on it is made (more) secure from external software attack and physical theft. Smart cards are analogous to lightweight versions of the TPM (without the physical theft prevention aspects).

The researchers looked into the silicon gate count of TPM. A silicon gate performs some logical operation (e.g., and, or, xor), and they are primarily

implemented electronically using transistors. Researchers found that implementing ECC does offer improvements in software performance, but ECC can be particularly efficient in reducing hardware workload due to increased efficiency. As computing environments move to trusted environments and hardware-based implementations of security functions, the benefits of ECC will increase dramatically in comparison to RSA. Optimized chip designs have been shown to be as much as 37 times faster than comparable implementations in software. As technology improves, the size of chip tends to shrink by a factor of ten (e.g., 3,260 gates compared to 34,000 gates as seen in Table 5). When optimized for speed, ECC is seven times faster when implementing current key lengths (1024-bit RSA at 2.6 ms vs. ECC-163 at 0.35 ms), and more than 80 times faster when using key lengths on the horizon for future security levels. (Zhang & Zhou & Zhuang & Li, 2007)

Algorithm	Optimization	Time	Gate count
RSA-1024 ECC-163	Space- optimized	4.90ms 0.66ms	34,000 3,260
RSA-1024 ECC-163	Speed- optimized	2.60ms 0.35ms	150,000 48,400
RSA-3072 ECC-283	Space- optimized	184ms 29ms	50,000 6,660
RSA-3072 ECC-283	Speed- optimized	110ms 1.3ms	189,200 80,100

Table 5. ECC and RSA gate counts [From Zhang & Zhou & Zhuang & Li 2007]

The Beijing Institute of Technology researchers concluded that elliptic curve cryptography is better suited to be used in TPM than the currently popular 2048-bit RSA because it is able to provide better performance during

signature and verification operations while appending less overhead to the certificate. Additionally it provides gate counts for ECC that are significantly smaller, which means lower chip costs. ECC requires fewer processor cycles, which allows the device to create less heat and ultimately less power drain. Finally, it requires less bandwidth for transactions due to more efficient protocols.

e. Sun Microsystems SSL Performance Study

Researchers at Sun Microsystems and Dogulus Stebila performed a number of experiments on replacing RSA with ECC in secure Web transactions. Table 6 shows that there is significant benefit to be gained from using ECC in SSL/TLS. ECC outperformed RSA by a factor of 2.4 measuring operations per second (1024-bit RSA and 160-bit ECC). Likewise, operations per second were improved by a factor of 11 using 2048-bit RSA and 224-bit ECC. Researchers used Apache 2.0.45 compiled with OpenSSL on a 900 MHz UltraSPARC 111. (Gupta & Stebila & Shantz, 2004)

	ECC-160	RSA-1024	ECC-224	RSA-2048
Ops/sec	271.3	114.3	195.5	17.8
Speed up	2.4 : 1		11 : 1	

Table 6. Secure Web transaction efficiency [From Gupta & Stebila & Shantz, 2004]

E. THREATS AND VULNERABILITIES

1. ECC Challenges

In 1997, Certicom Inc. issued the ECC challenge in order to demonstrate the security of ECC. By announcing a

list of elliptic curves and associated ECDLP parameters they set the stage, additionally offering a reward for solving the problem.

For ECC over prime fields GF(p), challenges have been defined for the bit sizes, k = {79, 89, 97, 109, 131, 163, 191, 239}. Certicom provided estimates for the required number of machine days for solving each challenge based on Intel Pentium 100 processors. The 160 bit and above, still requires a prodigious amount of computational power and is too costly to attack. Based on previous successful attempts to solve the ECDLP with smaller values for k, a successful attack against ECC-163 with a one year time limit would require 1.16X10^10 processors, which would cost on the order of $5.8X10^11. These staggering numbers are 2900 times larger than the cost of a special purpose hardware attack on 1024-bit length RSA. The ultimate conclusion is that ECC is as secure as had been commonly advertised. (Gueneysu & Paar & Pelzl, 2007)

F. KEY MEASURES OF EFFECTIVENESS/PERFORMANCE (MOE/MOP)

1. Key Efficiency

Elliptic curve cryptosystems are demonstrably more computationally efficient than the first generation public key systems currently in use. Although elliptic curve arithmetic is slightly more complex per bit than either RSA or DH arithmetic, the added "strength per bit" would seem to compensate for any extra compute time. The following table shows the ratio of DH computation versus EC computation for each of the key sizes listed in Table 2 of Chapter III. (National Security Agency Central Security Service, "The Case for Elliptical Curves", 2009)

Security Level (bits)	Ratio of DH Cost : EC Cost
80	3:1
112	6:1
128	10:1
192	32:1
256	64:1

Table 7. Relative Computation Costs of Diffie-Hellman and Elliptic Curves [From National Security Agency Central Security Service, 2009]

Until recently, the complexity of generating keys on a smart card was inefficient and often impractical. With ECC, the time needed to generate a key pair is so short that even a device with the very limited computing power of a smart card can generate a secure key pair, provided a good random number generator is available. This also means that the card personalization process can be streamlined for applications in which non-repudiation is important.

2. Key Strength

Cryptographic algorithms that provide security services are specified in the NIST 800-57 publication. Several of these algorithms are defined for a number of key sizes. NIST provides guidance for the selection of appropriate algorithms with the corresponding key sizes. It emphasizes the importance of acquiring cryptographic systems with appropriate algorithm and key sizes to provide adequate protection for the expected lifetime of the system as well as any data protected by that system during the expected lifetime of the data.

a. Comparable Algorithm Strengths

Different cryptographic algorithms provide different levels of strength, depending on the algorithm and the key size used. Two algorithms are considered to be comparable for their given key sizes if the amount of work needed to determine the keys is approximately the same using a given resource. The security of an algorithm for a given key size is traditionally described in terms of the amount of work it takes to try all keys for a symmetric algorithm with a key size of X that has no short-cut attacks. An algorithm that has a Y-bit key, but whose strength is comparable to an X-bit key of a different symmetric algorithm is said to provide X bits of security. An algorithm that provides X bits of security would, on average, take $T2^{(X-1)}$ of time to attack, where T is the amount of time that is required to perform one encryption of a plaintext value and comparison of the result against the corresponding cipher-text value.

Table 8 below provides comparable security strengths for the Approved algorithms.

Bits of security	Symmetric key algorithms	FFC (e.g., DSA, D-H)	IFC (e.g., RSA)	ECC (e.g., ECDSA)
80	2TDEA[19]	$L = 1024$ $N = 160$	$k = 1024$	$f = 160\text{-}223$
112	3TDEA	$L = 2048$ $N = 224$	$k = 2048$	$f = 224\text{-}255$
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256\text{-}383$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 384\text{-}511$
256	AES-256	$L = 15360$ $N = 512$	$k = 15360$	$f = 512+$

Table 8. Common Algorithm Strength Comparison [From NIST, 2007]

Column 1 indicates the number of bits of security provided by the algorithms and key sizes in a particular row. Column 2 identifies the symmetric key algorithms that provide the indicated level of security. Column 3 indicates the minimum size of the parameters associated with the standards that use finite field cryptography (FFC). Examples of such algorithms include DSA for digital signatures, and Diffie-Hellman (DH). L is the size of the public key, and N is the size of the private key. Column 4 indicates the value for k (which corresponds to the key size) for algorithms based on integer factorization. Finally, Column 5 indicates the range of f (the size of n, where n is the order of the base point G) for algorithms

based on elliptic curves that are specified for digital signatures. The value of f is commonly referred to as the key size.

3. Processing Overhead

Closely related to the key size of different public key systems is the channel overhead required to perform key exchanges and digital signatures over a network. The key sizes for public keys in Table 1 of Chapter III is also roughly the number of bits that need to be transmitted each way over a communications channel for a key exchange. Although in the case of ECC, there is one additional bit that needs to be transmitted in each direction, which allows the recovery of both the x and y coordinates of an elliptic curve point. (National Security Agency Central Security Service, 2009)

The difficulty of the ECDLP algorithm means that relatively strong security is possible with smaller key and certificate sizes. The smaller key size in turn means that less EEPROM is required to store keys and certificates and that less data needs to be passed between the card and the application allowing for shorter transmission times.

As smart card applications continue to require stronger and stronger security (via longer keys), ECC can continue to provide adequate security with fewer additional system resources. In other words ECC smart cards are capable of providing higher levels of security without increasing their cost. ECCs reduced processing times also contribute significantly to why ECC meets the smart card platform requirements so well. By comparison, other public key systems involve so much computation that a dedicated

hardware device known as a crypto coprocessor is often required. This crypto coprocessor not only takes up space, but adds about 20 to 30 percent to the cost of a chip (about three to five dollars towards the cost of each card). With ECC, the algorithm can be implemented in available ROM, so no additional hardware is required to perform strong, fast authentication. (Certicom, 1998)

Implementing public key cryptography in a smart card application poses numerous challenges. Smart cards present a combination of implementation constraints that other platforms do not. Constrained memory and limited computing power are two of them. Current DoD CAC cards in the field today have between about 1K to 6K of RAM, 64 to 144 kilobytes of EEPROM, and 16 to 32 kilobytes of ROM with the traditional 8 to 16 bit CPU typically clocked at Internal CPU clock up to 30 MHz with synchronous operation. Any additional requirements of memory or processing capacity increase the cost to an already cost sensitive card. Smart cards are also slow transmitters. It can only communicate a maximum of 255 bytes per Application Protocol Data Unit (APDU) transaction. An APDU is the communication unit between a smartcard reader and a smartcard. In order to achieve acceptable application speeds, data elements must be small (to limit the amount of data passed between the card and the terminal). While cryptographic services that are efficient in memory usage and processing power are needed to contain costs, reductions in transmission times are also needed to enhance usability.

Given the rigid constraints on processing power, parameter storage, and code space, as well as slow

input/output associated with smart cards implementation of public key cryptosystems has been associated with high-end cards, typically with both large memory configurations and cryptographic coprocessors. Certicom Incs ECC implementations enable the deployment of lower cost smart cards without compromising any of the required performance and security features. ECC smart cards would not require as much memory, nor would they require a cryptographic coprocessor to deliver strong authentication.

In Summary, ECC key size advantages afford many benefits for smart cards, and the superior performance available through judicious ECC implementations make applications feasible in low end devices without the need for additional dedicated crypto hardware. In channel-constrained environments, elliptic curves offer a much better solution than first generation public key systems.

G. CONCLUSIONS ON RSA AND ECC

In very general terms, elliptic curve cryptosystems offer the same security that an RSA system or a discrete logarithm based system offers but with significantly smaller key lengths. In terms of speed, however, it is quite difficult to give a quantitative comparison. This is partly due to the various optimization techniques that can be applied to different systems. Generally elliptic curve cryptosystems are faster than their corresponding discrete logarithm based systems. Elliptic curve cryptosystems are faster than the RSA system in signing and decryption, but are slower in signature verification and encryption. (RSA Security, 2004)

However, ECC implementation could have a significant impact on smaller devices such as PDAs or smart cards as the relative computational performance advantage of ECC over RSA is not indicated by key sizes but by the cube of the key sizes. The difference becomes even more dramatic as the increase in RSA key sizes leads to an even greater increase in computational cost. Going from 1024-bit RSA key to 3072-bit RSA key requires about 27 times as much computation while ECC would only increase the computational cost by just over 4 times (Vanstone, 2004).

V. IMPACT OF DOD MIGRATION TO NSA SUITE B

A. INTRODUCTION

Public key cryptography, using digital certificates, offers the best available technology for secure transmission of unclassified data across public and private networks. It provides a high degree of assurance of confidentiality, integrity, access control, and user identification among users of networked applications, including e-mail, Web-based information transactions, and other electronic commerce. The DoD PKI refers to the framework and services that provide for the secure generation, production, distribution, control, and accounting of DoD public key certificates. Its implementation was mandated by a DoD memorandum from the Department of Defense dated 6 May 1999, with a target completion date of October 2004.

Within this framework, DoD planners have worked to gain and maintain the initiative in the struggle against those entities that would seek to benefit from breaches of information security. With this in mind, the DoD and the U.S. Government as a whole face a paradigm shift from the classical discrete logarithm and integer factorization key generation techniques now in widespread use, to the newer, more creative methods of elliptic curve cryptography.

B. CAC ISSUANCE TIMES

One of the primary goals of the DoD PKI is to reduce the amount of time required for the CAC issuance process. Currently with RSA 1024 the typical issuance time falls within a three- to five-minute window. RSA 2048 is estimated to increase the time by an average of nearly two minutes. If this trend is followed (e.g., RSA 3072 and beyond) the issue times would continue to increase and perhaps become unacceptable. Here we come to one of the fundamental benefits of ECC over its predecessors.

As was shown in Chapter IV, CAC issuance using ECC will provide a significant reduction in processing time. According to the information contained in Table 3, 163-bit ECC takes 3.8 milliseconds to generate a key. From the RIM performance comparison in Table 4/ we see also that 256-bit ECC takes only 166 milliseconds. It is difficult to say with accuracy how long key generation will take using the limited resources on a CAC card, because RIMs BlackBerry is a faster, more capable platform compared to a typical smartcard chip, and because 163-bit ECC is not the size recommended by NIST. Regardless, it is clear that even without knowing the exact key generation time required for ECC, the general difference, which is an order of magnitude improvement, is significant.

From the CAC test lab studies at the DMDC mentioned in Chapter IV, we know RSA 2048 key generation times will be approximately 45 seconds per key. Multiply this number by three, which is the number of keys generated on a CAC (ID Certificate, PIV Authority Certificate, and Signature Certificate) and the reader will have a general idea of how much time each CAC requires.

Time savings will translate to higher productivity. The ability to produce more in less time will lead to customer satisfaction and ultimately lower costs. Consider the scenario where a large number of new CAC holders need to receive their cards — new contractors for example. A substantially smaller amount of time will be necessary to complete the CAC issuance process for each new individual.

C. EXISTING INFRASTRUCTURE

1. Current DoD PKI Architecture

According to the Deputy Secretary of Defense (DEPSECDEF) memorandum entitled "Department of Defense Public Key Infrastructure," as modified by the DoD Chief Information Officer (CIO) on 12 August 2000, there were at least twelve top-level milestones for the DoD PKI. The milestones included practical items like certificate issuance and registration infrastructure deployment. They also contained mission essential tasks like functional token certificate based access control and ensuring the ability to appropriately sign emails.

The time allocated to complete the year 2000 milestones was approximately four years. The authors of this thesis anticipate that the transition to NSA Suite B will be equally as comprehensive, but that the time required to fully integrate will not be as long as the original transition to PKI systems. This transition is expected to be difficult due mostly to factors of interoperability, which will be addressed to some degree by the modular architecture of the DoD PKI. (Information Assurance Technology Analysis Center, 2000)

The DoD PKI, which evolved from the DoD Medium Assurance Pilot PKI, supports the protection of business transactions and sensitive but unclassified administrative information. The DoD PKI can also be used on closed networks like SIPRNET or NIPRNET to provide additional protection such as user authentication and data separation. (Information Assurance Technology Analysis Center, 2000)

DoD PKI service employs a hierarchical architecture consisting of a centralized Root CA with a single level of subordinate CAs, a small number of Registration Authorities (RA), and a larger number of Local Registration Authorities (LRA). Currently LRAs function at RAPIDS stations, and certificates are able to be installed on DoD CACs. (Information Assurance Technology Analysis Center, 2000)

Given this existing infrastructure, a great deal of planning is necessary in order to implement a new cryptographic suite.

2. DoD PKI Hardware and Software

The DoD PKI PMO procures all hardware required for PKI implementation at the Root and CA levels. At the lower levels the various services or governmental agencies are responsible for acquiring, installing, accrediting, operating, and maintaining components associated with PKI. (Information Assurance Technology Analysis Center, 2000)

Currently, the DMDC has deployed integrated RAPIDS workstations at approximately 2,000 DoD RAPIDS stations at various locations around the world. These integrated RAPIDS workstations were meant to support the issuance of

the four certificates (only three of which are generated inside the DoD CAC) to all military, civilian, and selected contractor personnel.

Each RAPIDS station is manned with a Verification Officer (VO). All of these VOs will require updated training, as well as replacement CACs with updated ECC standards. CAC replacement is necessary because in order to issue ECC certificates the issuer must possess a sufficient security level. Additionally, all 2,000 workstations will need to be updated to support ECC. This process will be exacerbated by the fact that the stations are deployed around the globe.

a. Implementation at the Local Commands

Where applications employing public key technology are required, the local commands and DoD organizational elements are responsible for developing and deploying public key enabled applications (or integrating commercially available PK enabled products) that are compatible and compliant with the DoD PKI. In accordance with DEPSECDEF memorandum dated 12 August 2000, all the former PK enabled applications have been transitioned so that they are DoD PKI compliant. Additionally, all Web servers have been PK enabled to perform server-side authentication using SSL Authentication and DoD public key certificates. Currently, the vast majority of DoD Web servers are capable of performing client-side authentication using DoD CAC certificates. (DoD Public Key Infrastructure Program Management Office, 2000)

To date, all active duty military personnel, members of the Selected Reserve, DoD civilian employees,

and eligible contractor personnel who require access to DoD systems have received their PKI certificates on a DoD CAC (over 10 million since 2002). Although DMDC installed the upgraded integrated RAPIDS workstation at its RAPIDS locations, the responsibility for issuing DoD CACs remains with the DoD organizations operating the RAPIDS stations. For most services, the DoD RAPIDS stations are typically located at local Personnel Support Detachment (PSD), Pass & ID, or Security offices. Individual departments were left with the responsibility to develop specific plans and identify any additional resources needed to support certificate and CAC issuance. (DoD Public Key Infrastructure Program Management Office, 2000)

Here the impact of implementation of NSA Suite B is that the local commands will now be responsible for integration within their current command specific CAC enabled applications. This will be costly in terms of re-design, testing, development and integration. If the new methods do not work a hybrid solution will have to be looked at which will be discussed at length in Chapter VI. All of this will lead to potentially significant costs in time and money.

b. DoD Certification Authority

The agency responsible for all aspects of the DoD PKI is the DoD PKI Program Management Office (PMO). DoD PKI PMO coordinates all component and system development and testing through the efforts of its three working groups. The DoD PKI Technical Working Group is responsible for identifying, addressing, and resolving technical and operational issues associated with the implementation and

operation of the DoD PKI. The DoD PKI Business Working Group is responsible for addressing DoD PKI business requirements. The DoD PKI Certificate Policy Management Working Group (CPMWG) is responsible for preparing and coordinating the DoD Certificate Policy (CP), including the creation, review, and update of all relevant documentation.

The NSA, supported by DISA, serves as the DoD PKI PMO as directed by the Assistant Secretary of Defense for C4I, and provides coordination of activities across the DoD to define and implement the DoD PKI. The PMO provides the leadership and coordination for all PKI activities across the Department, and is the single point of responsibility for all DoD PKI planning, development, and implementation activities. The DoD PKI PMO is responsible for overall program management of all DoD efforts required to execute the DoD PKI transition. The PMO is also responsible for raising awareness of the status of ongoing and planned PKI related activities, and the support that is available for their effective use. (DoD Public Key Infrastructure Program Management Office, 2000)

The impact of migration at this level will be the scope of the broad planning efforts required. NSA will need to continue to liaison with DISA, and DISA will have to maintain communications with the DMDC in order to receive and deliver guidance and progress reports. In addition to the large scale coordination efforts required, the monetary costs associated with renewed licensing, third party consulting fees and the myriad other miscellaneous expenses that will arise will have an impact.

D. MIGRATION COSTS

In order to stay relevant the DoD PKI structure must continually evolve over time, and the PKI Program has established a fundamental philosophy for these transitions. Enhanced system capabilities must be introduced in parallel with existing operational capabilities. Every effort will have to be made to ease the operational impact to subscribers resulting from the evolution of infrastructure capabilities. Whenever feasible, the transition strategy should not be based on hard cutovers. This will allow subscribers to plan and implement effective transitions of their operations to take advantage of the newest capabilities.

The DoD PKI has adopted a highly modular, nodal architecture for the evolving DoD key management infrastructure. The architecture is built on four types of nodes. The first is the Client Node, which represents the subscribers that require products and services from the PKI. The next node is the Primary Services Node (PRSN). The PRSN is the core element of the PKI structure, providing common management functions in a server-based architecture. It offers client nodes access to the production sources, providing direct delivery of PKI products and services to applications that require them. It also handles subscriber access control and manages the interfaces between the other nodes. The third node is the Production Source Nodes (PSNs), which interfaces to the common management functions of the PRSN (e.g., the PKI CA that provides certificate management functions such as certificate creation, posting, rekeying, and revocation).

Finally, the Central Services Node (CSN) provides overall system management and configuration management functions for the infrastructure, including the long-term system archive and the master key management infrastructure database. The CSN also handles system health monitoring and overall infrastructure security management, including intrusion detection security oversight and audit data and analysis. (DoD Public Key Infrastructure Program Management Office, 2000)

By enforcing modularity while maintaining control of both physical and functional interfaces, PKI features and capabilities are theoretically set up to evolve over time in a structured and cost effective manner, which will facilitate an eventual cutover to Suite B.

1. DoD PKI 5.0

As indicated earlier, the DoD PKI evolution is designed to offer PKI products and services with a transition transparent to subscribers. The detailed design and planning for this evolution is extensive, but the evolutionary strategy is fairly straightforward.

The Medium Assurance PKI pilot was transitioned to the Class 3 PKI (Release 1.0) in April 1998. In July 2000, the DoD Class 3 PKI (Release 2.0), which introduced the use of newer certificates was approved for operational use. Efforts to incorporate PKI LRA functionality into RAPIDS terminals were completed. These updated RAPIDS terminals were introduced in Class 3 PKI Release 3.0 to provide a means for registering users enrolled in DEERS into the PKI and issuing CACs (smart cards) that serve as PKI hardware tokens. The Class 3 PKI Release 3.0 will also continue to

support certificates in software. PKI Release 4.0 provides an initial set of Class 4 (Defense Messaging System or DMS) PKI products and services consistent with those provided by the existing Class 3 PKI. (DoD Public Key Infrastructure Program Management Office, 2000)

The basic definition for DoD PKI Release 5.0 includes support for access control mechanisms that enables the transition of DMS organizational messaging subscribers to the DoD PKI. It will introduce an initial set of trusted date and trusted time stamp services. It will provide an initial capability for integrity/software download certificates. It will allow additional support for new Key Exchange and DSA algorithms (like ECC). Finally, it will provide toolkits for PKI-aware applications. (DoD Public Key Infrastructure Program Management Office, 2000)

a. Infrastructure Management

DoD PKI Release 5.0 will provide regional deployments of PKI Primary Service Nodes. It will allow for the ability to create new roles and dynamic mapping of privileges to those roles. It will create enhance existing PKI Help Desk features including an expanded repository of PKI information with on-line access available to authorized users. Its external interoperability will be expanded to approved Allied and Coalition partner PKIs. It will integrate Class 3 and 4 PRSN structures. Finally, it will incorporate an independent CSN with electronic access to all PRSNs. (DoD Public Key Infrastructure Program Management Office, 2000)

b. Anticipatory Developments

There are a number of additional PKI related activities that are substantial in order to accomplish the 5.0 Release. These activities are intended to ensure the smooth progression of PKI capabilities. Among them is the development of an audit reduction tool, implementation of the EC algorithm, and the development of a key management Applications Programming Interface (API). Additionally, time stamp application and processing, and prototype automated accounting and archive capabilities. Release 5.0 will also be used to enhance the tactical aspects of DoD PKI to include a tactical network model, protocol simulators and demand simulators. Release 5.0 will also act as an integration tool to ease into the Release 6.0 transition by merging a prototype deployable PRSN and a PSN simulator for new algorithms as well as prototype Class 5 PKI PRSN and PSN capabilities. As always, these prototyping activities are subject to the availability of funds and are subject to the priorities that are established at the time of their initiation. (DoD Public Key Infrastructure Program Management Office, 2000)

2. Money

a. Hardware Replacement

Most of the core infrastructure components associated with the PKI (i.e., RAs, directory components and LRAs) are already in place within the DoD. Currently LRA functional capability at the DoD RAPIDS stations is nominal. The oversight of the operation of LRAs, and issuance of CACs and Smart Card readers, via DEERS/RAPIDS is also already in place. Finally, additional LRAs beyond

those provided by DoD RAPIDS stations have been established and can provide RA/LRA operations, maintenance, and life cycle support (Information Assurance Technology Analysis Center, 2000).

The implementation of ECC itself will not require any significant hardware updates. Currently planned hardware refreshment plans will be sufficient and should remain unaffected by the migration. The only real hardware that will have to be replaced across the board is the actual cards that are issued to DoD personnel but even that will occur within the normal replacement cycles, which will be facilitated by the phased transition approach.

b. Infrastructure Upgrade

The PKI PMO has the task of identifying toolkits and ensuring they are available to facilitate the interaction and customer support for programs and vendors that are actually performing PKI transitions. DISA will lead activities to identify and evaluate the effectiveness of commercial PKI toolkits. NSA will take the lead for developing specialized toolkits needed to address specific requirements that cannot be satisfied from the private sector. DISA Joint Integration and Testing Command (JITC) has been established as a PK enabled applications test facility for developers and integrators to verify compliance and compatibility of their implementations within the PKI. All DoD services and agencies will retain the responsibility for enabling their applications and devices to be compatible with current PKI capabilities,

including the funding needed to maintain and operate the test facilities. (NIST Information Technology Laboratory, 2000)

The DMDC is responsible for the development and upgrade of RAPIDS terminals to incorporate the functionality and establish operations needed for them to serve as LRAs for the PKI. The DMDC would perform the RAPIDS development activities.

c. Refresh Plan

The PKI PMO will assume overall responsibility for procuring, or directing the procurement of all centrally operated infrastructure elements. The PKI PMO will develop the acquisition strategy for any significant changes to the DoD PKI. Concurrently, NSA and DISA will procure, develop, or direct the procurement of the centrally operated infrastructure elements of the key management infrastructure. DISA will also be responsible for the procurement and deployment of centralized directory elements of the PKI. The DoD PKI PMO will develop the acquisition strategy for the DoD PKI, the certificate management components and services as well as the refresh plan. DISA is the lead for the integration of the centralized components of the PKI, including the CA servers and directory components. The services and agencies will procure local infrastructure elements, PKI RA and LRA workstations and local directories.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. MANAGING RISK

A. INTRODUCTION

Assuming DoD will eventually wholly migrate to Suite B, risk management will play a large role in the successful transition. The purpose of this chapter is to identify risk and then make some educated recommendations on how to best mitigate those risks.

B. METHODOLOGY

Within this chapter, we intend to review the RSA 1024 to RSA 2048 Migration Plan. From this plan, we will evaluate the lessons learned. This information will then be factored into the creation of an original critical path analysis for the RSA to ECC transition.

Also within this chapter, we will identify some of the other potentially high risk areas associated with development, deployment and operation of NSA Suite B within the DoD framework. Recognizing that effective risk management is critical to the success of any program, we break it down into its two fundamental parts; assessment and mitigation.

C. NSA SUITE B MIGRATION RISKS

1. Proprietary Complications

As a way of clearing the way for the implementation of elliptic curves to protect US and allied government information, the National Security Agency purchased a license from Certicom Inc. that covers all of their intellectual property in a restricted field of use. The

license would be limited to implementations that were for national security uses and certified under FIPS 140-2 or were approved by NSA. Further, the license would be limited to only prime field curves where the prime was greater than 2255. On the NIST list of curves 3 out of the 15 fit this field of use: the prime field curves with primes of 256 bits, 384 bits and 521 bits. Certicom Inc. identified 26 patents that covered this field of use. NSAs license includes a right to sublicense these 26 patents to vendors building products within the restricted field of use. Certicom Inc. also retained a right to license vendors both within the field of use and under other terms that they may negotiate with vendors. (Certicom, 2000)

a. Local Level Challenges

As previously alluded to in Chapter V, local commands will be responsible for their command specific CAC enabled applications. This will present some unique challenges due to the newness of the technology. Local Commands will face application upgrade issues in terms of hiring subject matter experts who will be necessary to facilitate the software upgrades. Moreover, they may have to deal with the patent issues associated with Certicoms commercial ownership of ECC.

2. Software Compatibility

Less than a year prior to the publication of this thesis, Windows announced that its Vista Service Pack 1 and Windows Server 2008 would support Suite B cryptographic algorithms as a part of its Cryptography Next Generation (CNG). For Windows 7.0 and Server 2008 R2, TLS and

Encrypting File System (EFS) will be implemented using Suite B algorithms. Currently CNG is not FIPS 140-2 level 2 certified, nor is it Common Criteria certified, which prohibits its use within the DoD. FIPS 140-2 precludes the use of un-validated cryptography for cryptographic protection of sensitive data within the federal system. Invalidated cryptography is viewed by NIST as providing no protection to the information or data – in effect the data would be considered unprotected plaintext. If the agency specified that information or data be cryptographically protected, then FIPS 140-2 is applicable, and if cryptography is required, it must be validated. (NIST Information Technology Laboratory, 2009)

With the passage of the Federal Information Security Management Act (FISMA) of 2002, there is no longer a statutory provision to allow agencies to waive mandatory FIPS. The waiver provision had been included in the Computer Security Act of 1987; however, FISMA superseded that act.

Although the latest versions of Windows applications are in the process of incorporating Suite B for use within the DoD, earlier versions will not be retroactively upgraded. For example, Windows XP and Server 2003 will not be supported and will eventually have to be phased out of use as Suite B incrementally supplants older methods.

3. RSA Critical Path Analysis

We will be using the Critical Path Method (CPM) or Critical Path Analysis to analyze former DoD migration projects to provide perspective on the migration to Suite B. We will also create Network Planning Diagrams, which

are very effective planning tools for unique projects that contain interactions between several components with many interrelated tasks. CPM is also an effective procedure for using network analysis to identify those tasks on the critical path, which have the potential to lengthen the overall project timescale. For most deviations from the critical path (late starts, early starts, etc.) there is a degree of tolerance within which project success is still likely.

a. RSA 1024 to RSA 2048 Lessons Learned

The major tasks that must be completed on time for the migration of RSA 1024 to RSA 2048 are outlined in Table 9 below.

Description		Required Predecessor	Duration (Weeks)
1.	Develop Migration Process for Software		1
2.	CAC Enable Application Testing		21
3.	CA Upgraded		38
4.	Sample Alpha Cards Received		27
5.	Go Through FIPS Certification		
6.	Update CAC Infrastructure Software	1	17
7.	Server Test Upgrade	1	21
8.	RAPIDS Upgrade	1	20
9.	Applet/Card Integration Test	4	2
10.	Infrastructure software Install / Patch update	6	20
11.	CAC Enable Application Test	8	6
12.	Provide Server Certificates	3	1
13.	Test CA	3	5
14.	CAC Infrastructure Integration Test	10	1
15.	Deploy New Server	15	2
16.	Install Server Certs	12	2

17.	Alpha Card Internal Testing	9	11
18.	Alpha Card Customer Testing	17, 12	4
19.	CAC Infrastructure Deploy to Test Environment	14	1
20.	CAC Issuance Integration Test	15, 11, 16, 19	2
21.	Order Beta Cards	9	13
22.	Performed new Card Keys Ceremony	21	2
23.	Issue Beta Card	20, 13, 22	1
24.	Beta Card Testing	23	9
25.	Approve for Production	24, 5	1

Table 9. Critical Tasks for RSA 1024 to RSA 2048

From this table a network activity diagram was created that shows the dependent sequence of activities. In the diagram, a network of tasks is set up to show which need completion before other tasks can be started. This helps to identify the critical path, which is the route through the network that will take the most time. If there are more than one predecessor tasks, then there will be several possible early starts. The largest of these is the most important. The early finish for each task is equal to the early start plus the task duration. The final calculation is for the earliest completion time for the project. This is calculated like the early start date. Starting with the tasks at the end of the diagram, calculate the late start and late finish for each task in turn, following the arrows in the reverse direction, as in Figure 11 below.

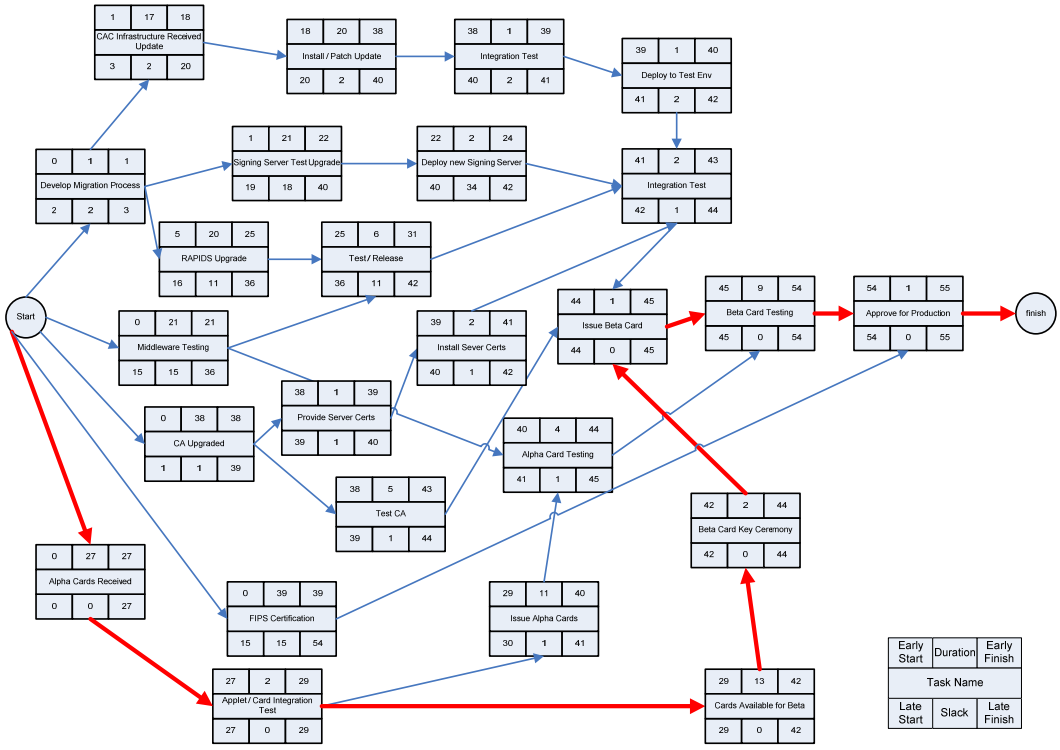

Figure 11. RSA 1024 to RSA 2048 Critical Path

The late finish is the same as the late start of the succeeding task (for the final tasks in the project, this is equal to the earliest completion date). If there is more than one successor task, then there are several possible late tasks. We select the smallest of these. The late start for each task is the late finish minus the task duration. The final calculation is for the earliest completion time for the project. This is calculated in the same way as the early start date. We calculate slack time by subtracting the early start from the late start. The slack time is the amount of time the task can be slipped without affecting the end date. Tasks on the critical path have no slack.

Task Name	Description	Required Predecessor	Duration (Weeks)	Early Start (ES)	Early Finish (EF)	Late Start (LS)	Late Finish (LF)	Slack
1.	Develop Migration Process for Software		1	0	1	2	3	2
2.	CAC Enable Application Testing		21	0	21	15	36	15
3.	CA Upgraded		38	0	38	1	39	1
4.	Sample Alpha Cards Received		27	0	27	0	27	0
5.	Go Through FIPS Certification		39	0	39	15	54	15
6.	Update CAC Infrastructure Software	1	17	1	18	3	20	2
7.	Server Test Upgrade	1	21	1	22	19	40	18
8.	RAPIDS Upgrade	1	20	5	25	16	36	11
9.	Applet/Card Integration Test	4	2	27	29	27	29	0
10.	Infrastructure software Install / Patch update	6	20	18	38	20	40	2
11.	CAC Enable Application Test	8	6	25	31	36	42	11
12.	Provide Server Certificates	3	1	38	39	39	40	1
13.	Test CA	3	5	38	43	39	44	1

14.	CAC Infrastructure Integration Test	10	1	38	39	40	41	2
15.	Deploy New Server	15	2	22	24	40	42	34
16.	Install Server Certs	12	2	39	41	40	42	1
17.	Alpha Card Internal Testing	9	11	29	40	30	41	1
18.	Alpha Card Customer Testing	17, 12	4	40	44	41	45	1
19.	CAC Infrastructure Deploy to Test Environment	14	1	39	40	41	42	2
20.	CAC Issuance Integration Test	15, 11, 16, 19	2	41	43	42	44	1
21.	Order Beta Cards	9	13	29	42	29	42	0
22.	Performed new Card Keys Ceremony	21	2	42	44	42	44	0
23.	Issue Beta Card	20, 13, 22	1	44	45	44	45	0
24.	Beta Card Testing	23	9	45	54	45	54	0
25.	Approve for Production	24, 5	1	54	55	54	55	0

Table 10. Critical Tasks for RSA to ECC

4. ECC Critical Path Risk Analysis

Based on the nature of the risks identified earlier, an estimate of the duration of each activity associated with Suite B migration is possible. The activities were each given three time estimates. In so doing, uncertainty in completion time is accounted for to an acceptable degree. This is accomplished by using the following formula:

$$t_e=((t_0+4(t_m)+t_p))/6$$

Optimistic Time (t_0) is the time in which any particular activity may be completed if everything goes well and there are no complications. Most Likely Time (t_m) is the time in which a particular activity can most often be completed under normal conditions. If an activity is repeated many times, the duration of time to accomplish this activity that occurs most often would be equivalent to the most likely time estimate. Pessimistic time (tp) is the time in which a particular activity may be completed under adverse conditions, such as having unusual and unforeseen complications. (Gido, 1985)

It may be possible to reduce the critical path of a project (and consequently pull in the completion date) by rearranging some tasks, which have an optional sequence or by moving key personnel into tasks in the critical path.

Our prediction for the migration from RSA to ECC is that the lack of applications that are ready for Suite B implementation and the lack of completed field testing will be a major factor in the prolonging of the completion of Suite B migration.

Figure 12 and Table 11 below illustrate the related critical path and essential tasks.

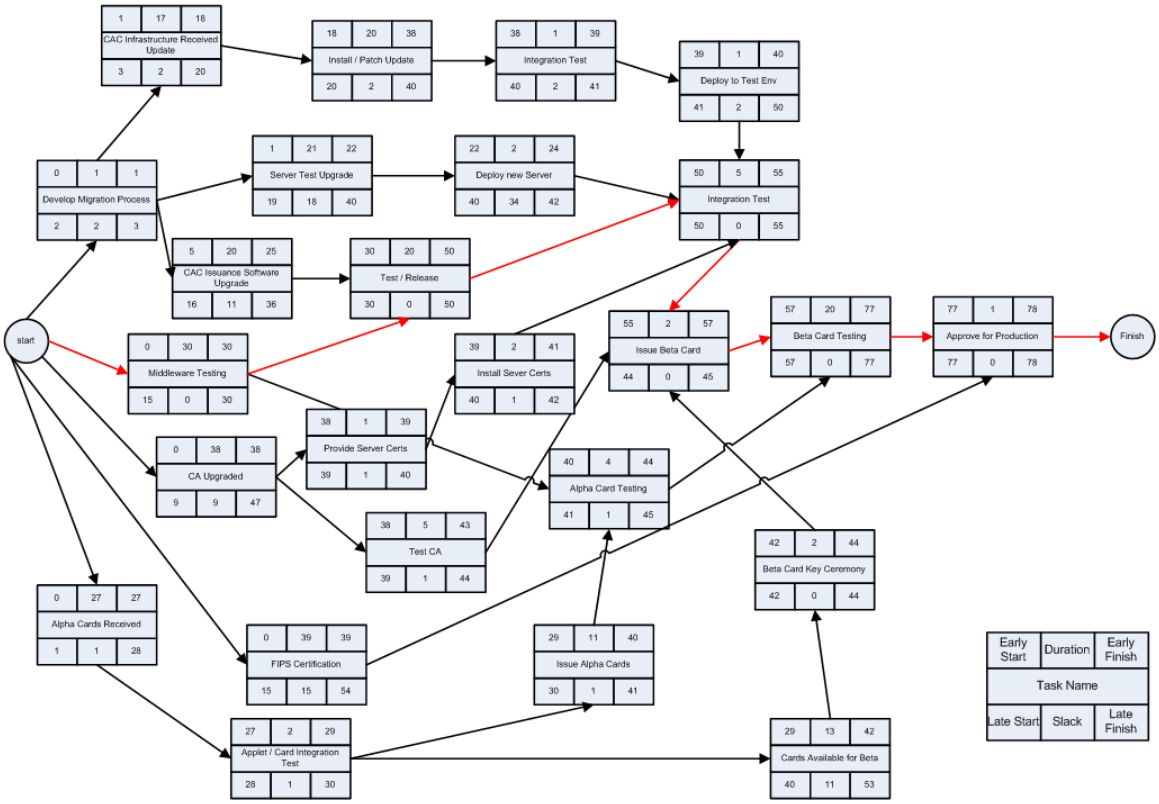

Figure 12. RSA to ECC Critical Path

Task Name	Description	Duration (Weeks) RSA	te	to	tm	tp	Early Start (ES)	Early Finish (EF)	Late Start (LS)	Late Finish (LF)	Slack
1.	Develop Migration Process for Software	1	2.00	1	2	3	0	2	9	10	9
2.	CAC Enable Application Testing	21	30.17	21	30	40	0	30	15	36	15
3.	CA Upgraded	38	38.00	36	37	44	0	38	9	47	9
4.	Sample Alpha Cards Received	27	27.17	26	27	29	0	27	1	28	1
5.	Go Through FIPS Certification	39	39.33	38	39	42	0	39	38	77	38
6.	Update CAC Infrastructure Software	17	17.33	15	17	21	1	18	11	28	10

7.	Server Test Upgrade	21	21.17	19	21	24	1	22	27	48	26
8.	RAPIDS Upgrade	20	20.17	18	20	23	5	25	10	30	5
9.	Applet/Card Integration Test	2	2.33	2	2	4	27	29	28	30	1
10.	Infrastructure software Install / Patch update	20	20.00	19	20	21	18	38	28	48	10
11.	CAC Enable Application Test	6	20.00	15	20	25	30	50	30	50	0
12.	Provide Server Certificates	1	1.17	1	1	2	38	39	47	48	9
13.	Test CA	5	5.17	4	5	7	38	43	50	55	12
14.	CAC Infrastructure Integration Test	1	1.17	1	1	2	38	39	48	49	10
15.	Deploy New Signing Server	2	2.17	2	2	3	22	24	48	50	26
16.	Install Server Certs	2	3.00	2	3	4	39	42	39	42	0
17.	Alpha Card Internal Testing	11	11.17	10	11	13	29	40	30	41	1
18.	Alpha Card Customer Testing	4	4.33	3	4	7	40	44	53	57	13
19.	CAC Infrastructure Deploy to Test Environment	1	2.00	1	2	3	39	41	49	50	10
20.	CAC Issuance Integration Test	2	5.00	4	5	6	50	55	50	55	0
21.	Order Beta Cards	13	13.17	12	13	15	29	42	40	53	11
22.	Performed new Card Keys Ceremony	2	2.33	2	2	4	42	44	53	44	11
23.	Issue Beta Card	1	2.00	1	2	3	55	57	55	57	0
24.	Beta Card Testing	9	20.00	9	19	35	57	77	57	77	0
25.	Approve for Production	1	1.33	1	1	3	77	78	77	78	0

Table 11. RSA to ECC Critical Tasks

D. RISK MITIGATION

1. Incremental Implementation

Much of the required infrastructure that supports everyday applications within DoD (e.g., Kerberos, Smart Card logon, S/MIME) is not fully Suite B supported. To date one of the few major protocols that are fully

supported is TLS/SSL (Microsoft, 2009). That being the case, TLS is an ideal place to start the incremental transition.

According to Bob Lord, Senior Engineering Director at Redhat TLS is primed for initial migration activities because it provides the fewest unpredictable variables. Specifically, servers are under the direct control of their owners. Additionally some newer browsers are already ECC enabled. By comparison, clients are more widespread and have many more varieties of requirements and policy constraints, which make them unlikely candidates to be a starting point. (Lord, 2009)

2. Software Upgrade Cycles

As a best practice for maintaining the highest levels of security, it is recommended that the latest version of a given browser (e.g. Internet Explorer, Firefox) is used for all Web-based applications.

3. Pilot Programs

When top-level changes are implemented — like the cryptographic migration of RSA to ECC — it is always prudent to start with a relatively small, controlled environment in which to test the new technology so that there is less risk of unforeseen consequences.

The lessons learned from pilot programs can be invaluable. Mistakes that occur on a relatively small scale can prevent system wide failures that would have more severe consequences.

The authors of this thesis highly recommend an ECC pilot program similar to the Medium Assurance Pilot program that led to the eventual implementation of DoD PKI.

4. Resource/Funding Allocation

Funding availability is always a potential risk when dealing with major government programs. Much of the responsibility for ensuring that PKI related programs receive their funding will fall to the PKI PMO. The PMO must coordinate with the various agencies (NSA, DISA, etc.) and services (USA, USMC, USN, USAF, etc.) involved to identify resources needed to complete the development of the architecture, perform security analysis and testing as well as procurement. It must also coordinate to identify funding and resources needed to deploy and operate the local infrastructure elements. All of these tasks will go towards ensuring that funding is effectively allocated for specific PKI related activities like migration to NSA Suite B.

5. Near Term Migration Path

a. Legacy Systems

Some legacy systems will not be able to make the move to ECC due mostly to the fact that there are too many deployed clients. This issue of deployment will prevent a quick upgrade to ECC. Many clients are not under direct control because they fall under the local command authority. When local priorities, logistics, budgets, schedules and overall lack of resources are considered, complete upgrades will often become easy targets for commanders looking to balance multiple needs. For example,

The U.S. Navys fleet used Windows NT long after the rest of the DoD had transitioned to Windows 2000/XP because widespread compatibility issues on sensitive systems. A similar decision now would have significant implications as the DoD transitions to NSA Suite B.

b. Solutions

DoD users will more than likely operate within a hybrid of ECC and RSA for the near term and into the foreseeable future. The modular nature of the PKI infrastructure as well as the phased implementation approach will drive this hybrid solution as much as the fact that schedules are almost certain to be extended to accommodate delays. Among the implications of this hybrid approach are the need for servers that can support both ECC and RSA.

Web servers like Fortitude (Netscape replacement) support both ECC and RSA on one platform. In this fashion, new ECC enabled clients can still talk to old RSA-only clients via use of the same server. Moreover, this will enable a smooth transition well in advance of any final hard RSA cutoff.

VII. CONCLUSIONS

A. INTRODUCTION

Public key cryptography has become a mainstay for secure communications over the Internet and throughout many other forms of communication. It provides the foundation for key management and digital signatures. With key management, it is used to distribute the secret keys used in other cryptographic algorithms. Regarding digital signatures, it is used to authenticate the origin of data and protect the integrity of that data. It is paramount to the security of the United States of America that the information deemed too sensitive to be viewed by antagonistic entities remains secure. It is from this point of view that the magnitude of the implications of a failed transition to new cryptographic techniques is fully realized.

B. FINDINGS

The Authors of this thesis have attempted to determine some of the potential impacts of the DoD migration to NSA Suite B on CAC usage, issuance and performance. Testing was divided into two parts. The first was a comparison of 1024- and 2048-bit RSA key generation and Secure Sockets Layer (SSL) authentication, including detailed statistical information on the difference between the 1024-bit keys and the 2048-bit keys. We found a significant delta in key generation times from RSA 1024 to RSA 2048 although there was little noticeable difference when comparing encryption and decryption times and digital signature generation times.

That said, if the DoD continues its present course of action of further increasing RSA key lengths, the next iteration will be 3072 bits. A key length of that size would be unsustainable in terms of smart card issuance for the DoD, given that 2048-bit keys are already taking nearly two minutes longer to generate per user than RSA 1024. As noted in Chapter IV, the relative computational performance advantages of ECC over RSA are compelling. Prominent telecommunications company, Research In Motion, has also stated publicly that the key pair generation for RSA 3072 is "too long" even for their platforms, which have greater computing resources than a smart card.

Additionally, based on findings obtained through DMDC lab testing of CAC cards using different RSA key lengths as well as independent private sector testing of ECC, we assessed that Elliptic Curve Cryptography will provide comparable security with more efficient performance than the first generation public key techniques currently in use. From this, the authors have determined that an attractive course of action is to implement the ECC alternative.

The authors identified and attempted to mitigate some of the risks associated with a DoD-wide migration to NSA Suite B. We identified the following areas as potentially highly risky: intellectual property, software compatibility, local level challenges, resources, funding, and scheduling. Unless these major risks are attended to, the Suite B migration project will be in jeopardy.

Risk mitigation notwithstanding, it will be a long while before the U.S. Government will be able to completely

transition to a new cryptographic suite of algorithms. An RSA-ECC hybrid solution is more likely as the PKI infrastructure slowly adapts to its changing cryptographic environment. In fact, the phased implementation mentality that has so far defined the implementation of PKI since its inception in the late 1990s will almost certainly demand a parallel solution that features both the old and the new for quite some time.

C. SUGGESTIONS FOR FUTURE RESEARCH

As was alluded to in Chapter IV, Smart cards that are enabled with ECC implementation are not currently available. When they are, more complete testing will be possible. This testing should include taking a measure of on-card performance of ECC. Additionally, more comprehensive testing of end-to-end ECC performance across DoD networks would be beneficial.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Aciicmez, O. & Koc, C.K. & Seifert, J.P. (2007). *On the Power of Simple Branch Prediction Analysis.* Proceedings of the 2nd ACM symposium on information, computer and communications security, p. 312-320.

Barker, E. & Johnson, D. & Smid, M. (March 2007). *Recommendation for pair-wise key establishment schemes using discrete logarithm cryptography (Revised).* NIST Special Publication 800-56A.

Barker, E. & Barker, W. & Burr, W. & Polk, W. & Smith, M. (March 2007). *Recommendation for key management-pt 1: general (revised). NIST* Special Publication 800-57.

Boneh, D. & Lipton, R.J. (1996). Algorithms for black-box fields and their application to cryptography. *Advances in Cryptology: Crypto '96*, 283–297.

Certicom (1998). *The elliptic curve cryptosystem for smart cards* [White paper].

Certicom Research (September 20, 2000). *Standards for efficient cryptography (SEC) v1.0.* "SEC1: Elliptic Curve Cryptography."

Certicom (2004). *The use of public key cryptography in Blackberry.* C0de and C1pher Vol. 2 no. 1: Certioms Bulletin of Security and Cryptography. Retrieved on June 2, 2009, from http://www.estig.ipbeja.pt/~rmss/passa/efi/v2i1-codeandcipher2-1.pdf

Cramer, R. & Shoup V. (1998). "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack." *Lecture Notes in Computer Science 1462*, pp. 13-25.

Daswani, N. (n.d). *Cryptographic execution time for WTLS handshakes on Palm os devices* [White paper]. Retrieved on May 29, 2009, from Stanford University: http://infolab.stanford.edu/~daswani/papers/WTLSPerformancePaper3.pdf

DeClercq, January (2006). *Symmetric vs. asymmetric ciphers.* Windows IT Pro,. Retrieved April 20, 2009, from http://windowsitpro.com/Articles/ArticleID/93787/pg/1/1.html

DoD Public Key Infrastructure Program Management Office (December 18, 2000). *Public key infrastructure roadmap for the department of defense version 5.0.* ASD(C3I)

Fivemack (2007, December 24). "518-bit GNFS with msieve" [MSG 1] Message retrieved May 2, 2009. Posted to: http://www.mersenneforum.org/showthread.php?t=9787

Gido, J. (1985). *An introduction to project planning*, Second Edition. Industrial Press Inc., 200 Madison Avenue, New York, NY 10157. pp. 24-27.

Gupta, V. & Stebila, D. & Shantz, S. (2004). *Integrating elliptic curve cryptography into the webs security infrastructure*. Retrieved on April 12, 2009, from http://research.sun.com/sunlabsday/docs.2004/p915-gupta-final.pdf

Information Assurance Technology Analysis Center (2000, November 29). *Public key infrastructure implementation plan for the department of the navy.*

Jallad, K. & Katz, J. & Schneier, B. (2003). *Implementation of Chosen-Ciphertext Attacks against PGP and GnuPG.* Retrieved April 30, 2009, from http://www.schneier.com/paper-pgp.pdf

Johnson, D. & Menezes, A. & Vanstone, S. (2001). *The Elliptical Curve Digital Signature Algorithm* [White paper]. Retrieved from Certicom Inc.

Johnson, D. (1996, July 16). *Diffie-Hellman key agreement small subgroup stack. A Contribution to X9F1. Certicom.*

Kocher, Paul C. (n.d.) *Timing Attacks on Implementation of Diffie-Hellman, RSA, DSS and Other Systems.* [White paper] San Francisco, CA: Cryptography Research, Inc.

Lenstra, A.K., & Verheul, E.R. (2000). *Selecting Cryptographic Key Sizes*, Proceedings PKC 2000, Lecture Notes in Computer Science, 7-11.

Lim, C.H. & Lee, P.J. (1997). *A key recovery attack on discrete log-based schemes using a prime order subgroup.* Advances in Cryptology: Crypto '97, 249-263.

Lord, B. (2008, April 28). *ECC interoperability*. Identity Protection and Management Conference [PowerPoint slides].

Menezes, A. & Okamoto, T. & Vanstone, S. (1990, September). *Reducing elliptic curve logarithms to logarithms in a finite field*, Unpublished manuscript.

Microsoft Technet (2009). *Cryptography next generation: How should I prepare for CNG?* Retrieved on 2 June 2009, from http://technet.miscrosoft.com/en-us/library/cc730763(printer).aspx

Montgomery, Peter L. (2008). *Preliminary Design of Post Sieving Processing for RSA 768*. Microsoft Research, Presented at CADO Integer Factorization Workshop.

National Security Agency Central Security Service (2009, January 15). *NSA Suite B Cryptography*. Retrieved May 1, 2009, from http://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml

National Security Agency Central Security Service (2009, January 15). *The case for elliptic curve cryptography*. Retrieved May 9, 2009, from http://www.nsa.gov/business/programs/elliptic_curve.shtml

NIST Information Technology Laboratory (2009, May 29). *Cryptographic module validation program (CMVP)* [Fact Sheet]. Retrieved June 1, 2009, from http://csrc.nist.gov/groups/STM/cmvp/

RSA Security (1998). *Frequently Asked Questions About Todays Cryptography* (2000), 4.1. In RSA Laboratories. Retrieved April 23, 2009, from http://www.rsa.com/rsalabs/node.asp?id=2261

RSA Security (2001, July 16). *RSA Security Announces Support for Department of Defense Common Access Card.* Press release. Retrieved April 25, 2009, from http://www.rsa.com/press_release.aspx?id=943

RSA Security (2009). *RSA Encryption and Key Management Suite*. Retrieved April 30, 2009, from http://www.rsa.com/node.aspx?id=1203

Stewart, B. *Public Key Cryptography (PKC) History*. Retrieved April 20, 2009, from http://www.livinginternet.com/i/is_crypt_pkc_inv.htm

U.S. Department of Commerce/National Institute of Standards and Technology (2000, January 27). *Digital Signature Standards (DSS)*. Federal Information Processing Standards Publication (FIPS) Pub 186-2.

U.S. Department of Commerce/National Institute of Standards and Technology (2002, August 1). *Secure hash algorithm*. Federal Information Processing Standards Publication (FIPS) Pub 180-2.

Vanstone, S. (1992) "Responses to NISTs Proposal," *Communications of the ACM*, 35, pp. 50-52.

Vanstone, S. (2004, March 18). *ECC holds key to next-gen cryptography*. D&R Industry Articles. Retrieved on 5 June 2009, from http://www.commsdesign.com/showArticle.jhtml?articleID=18400497

Zhang, x. & Zhou, M. & Zhuang, J. & Li, J. (2007). *Implementation of ecc based trusted platform module*. Proceedings of the 6th International Conference on Machine Learning and Cybernetics, Hong Kong, 19-22 August 2007

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Marine Corps Representative
   Naval Postgraduate School
   Monterey, California

4. Director, Training and Education, MCCDC, Code C46
   Quantico, Virginia

5. Director, Marine Corps Research Center, MCCDC, Code C40RC
   Quantico, Virginia

6. Marine Corps Tactical Systems Support Activity (Attn: Operations Officer)
   Camp Pendleton, California

7. Chairman, Information Sciences Department
   Naval Postgraduate School
   Monterey, California