WEST VIRGINIA
SECRETARY OF STATE
NATALIE E. TENNANT
ADMINISTRATIVE LAW DIVISION

Form #2

Do Not Mark In this Box

2011 JUN -8 PM 3:09

OFFICE OF WEST VIRGINIA SECRETARY OF STATE

NOTICE OF A COMMENT PERIOD ON A PROPOSED RULE

AGENCY: Secretary of DHHR; Insurance Commissioner; and Chair of the Health Care Authority

TITLE NUMBER: 114A

RULE TYPE: Joint Legislative Rule CITE AUTHORITY W. Va. Code §33-4A-8

AMENDMENT TO AN EXISTING RULE: YES____ NO X

IF YES, SERIES NUMBER OF RULE BEING AMENDED: _____

TITLE OF RULE BEING AMENDED: _____

IF NO, SERIES NUMBER OF RULE BEING PROPOSED: 2

TITLE OF RULE BEING PROPOSED: All-Payer Claims Database Program's Privacy and Security Rule

IN LIEU OF A PUBLIC HEARING, A COMMENT PERIOD HAS BEEN ESTABLISHED DURING WHICH ANY INTERESTED PERSON MAY SEND COMMENTS CONCERNING THESE PROPOSED RULES. THIS COMMENT PERIOD WILL END ON July 8, 2011 AT 5:00 p.m. . ONLY WRITTEN COMMENTS WILL BE ACCEPTED AND ARE TO BE MAILED TO THE FOLLOWING ADDRESS:

Timothy Murphy, Attorney Supervisor
WV Offices of the Insurance Commissioner
P. O. Box 50540
Charleston WV 25305-0540
Timothy.Murphy@wvinsurance.gov

THE ISSUES TO BE HEARD SHALL BE LIMITED TO THIS PROPOSED RULE.

Charles O. Lorensen
Cabinet Secretary
West Virginia Department of Revenue

ATTACH A **BRIEF** SUMMARY OF YOUR PROPOSAL

Department of Revenue
Agency Questionnaire

Re: Joint Legislative Rule to be Filed

TITLE 114A, SERIES 2
ALL-PAYER CLAIMS DATABASE PROGRAM'S PRIVACY AND SECURITY RULE

Question 1: Are regulations required?

Yes, W. Va. Code §33-4A-8 requires that rules be promulgated before any actions to collect data or to assess fees can occur. The APCD program involves compliance with data submission requirements and attendant penalties for non-compliance, as well as mandating the form of such submissions, so a rule is necessary to provide adequate notice to affected submitters (insurance carriers and TPAs) on these new requirements.

Question 2: Is the rule you are proposing controversial? If yes, what are the pros and the cons?

Possibly, although all stakeholders were fully engaged in the subcommittee that drafted the bill (SB&I) and have voiced no particular concerns. Much depends on the particular submission requirements and what will be demanded of the carriers. Highmark has recently noted some concerns with the rule and the program in general, particularly how the privacy of data will be protected.

Question 3: Is the rule you are proposing a copy of another state's rule? A model rule? Custom-drafted?

Custom drafted, although it borrows from the laws of other states with APCDs. The data submission specs will be those being developed by a national group.

Question 4: What are the really important things you think the Secretary of the Department of Revenue should know about this rule and the issues that surround it?

Although the stakeholders (carriers, providers) seem to be comfortable with the bill and with the concept of APCDs generally, there are concerns with ensuring the non-disclosure of confidential data.

Insurance Commissioner
Secretary of DHHR
Chair of Health Care Authority
Joint Legislative Rule
Title 114A, Series 2

ALL-PAYER CLAIMS DATABASE PROGRAM'S PRIVACY AND SECURITY RULE

TITLE 114A, SERIES 2

BRIEF SUMMARY OF RULE

This is a joint rule proposed pursuant to HB 2745 (RS 2011, effective June 10), which provides that the Insurance Commissioner, Secretary of DHHR and the Chair of the Health Care Authority (collectively, the "MOU parties") "shall execute" an MOU to develop an all-payer claims. The statute assigns to each of the MOU parties an area of primary responsibility: The Insurance Commissioner is primarily responsible for the collection of the data from insurers (series 1) and HCA is primarily responsible for the release of data to third parties. W. Va. Code §33-4A-2. This proposed rule addresses only the privacy and security provisions of the All-Payer Claims Database Program, an area of responsibility assigned to DHHR.

Insurance Commissioner
Secretary of DHHR
Chair of the Health Care Authority
Joint Legislative Rule
Title 114A, Series 2

ALL-PAYER CLAIMS DATABASE PROGRAM'S PRIVACY AND SECURITY RULE

TITLE 114A, SERIES 2

STATEMENT OF CIRCUMSTANCES

HB 2745, enacted during the 2011 regular session of the Legislature, provides that the Insurance Commissioner, Secretary of DHHR and the Chair of the Health Care Authority (collectively, the "MOU parties") "shall execute" an MOU to develop an all-payer claims database and provides that these agencies may propose joint legislative rules "as are necessary to implement" the all-payer claims database program. See W. Va. Code §33-4A-8. W. Va. Code §33-4A-8 further provides that no claims data may be collected and no fees may be assessed to users of such data until rules are promulgated. The statute assigns to each of the MOU parties an area of primary responsibility; the Insurance Commissioner is primarily responsible for rules and enforcement related to the collection of the data from insurers, DHHR for maintenance of the data and for the release of data to other agencies, researchers, etc. W. Va. Code §33-4A-2. This proposed rule addresses only the privacy issues involved in collecting and retaining the data.

APPENDIX B

FISCAL NOTE FOR PROPOSED RULES

Rule Title: All-Payer Claims Database Program's Privacy and Security Rule (114ACSR2)

Type of Rule: __X__ Joint Legislative _____Interpretive ____Procedural

Agency: WV Offices of the Insurance Commissioner

Address: Post Office Box 50540

1124 Smith Street, Greenbrooke Building

Charleston, West Virginia 25305-0540

Phone Number: (304) 558-6279 x1210 Email: Timothy.Murphy@wvinsurance.gov

Fiscal Note Summary

Summarize in a clear and concise manner what impact this measure will have on costs and revenues of state government.

See Fiscal Note for Title 114A, Series 1; this rule deals with privacy of the data.

Fiscal Note Detail

Show over-all effect in Item 1 and 2 and, in Item 3, give an explanation of Breakdown by fiscal year, including long-range effect.

FISCAL YEAR			
Effect of Proposal	Current Increase/Decrease (use "-")	Next Increase/Decrease (use "-")	Fiscal Year (Upon Full Implementation)
1. Estimated Total Cost	0	0	0
Personal Services	0	0	0
Current Expenses	0	0	0
Repairs & Alterations	0	0	0
Assets	0	0	0
Equipment	0	0	0
Other	0	0	0
2. Estimated Total Revenues	0	0	0

Rule Title: All-Payer Claims Database Program's Privacy and Security Rule (114ACSR2)

3. Explanation of above estimates (including long-range effect):
Please include any increase or decrease in fees in your estimated total revenues.

See Fiscal Note for Title 114A, Series 1; this rule deals with privacy of the data.

MEMORANDUM

Please identify any areas of vagueness, technical defects, reasons the proposed rule **would not** have a fiscal impact, and/or any special issues **not** captured elsewhere on this form.

See Fiscal Note for Title 114A, Series 1; this rule deals with privacy of the data.

Date: 6-8-11

Signature of Agency Head or Authorized Representative

Timothy Murphy, Associate Counsel
WV Offices of the Insurance Commissioner
P. O. Box 50540
Charleston WV 25305-0540
Timothy.Murphy@wvinsurance.gov

TITLE 114A
LEGISLATIVE RULE
ALL-PAYER CLAIMS DATABASE PROGRAM - SECRETARY, DEPARTMENT OF HEALTH AND HUMAN RESOURCES, COMMISSIONER, OFFICE OF THE INSURANCE COMMISSIONER AND CHAIR, WEST VIRGINIA HEALTH CARE AUTHORITY

SERIES 2
ALL-PAYER CLAIMS DATABASE PROGRAM'S PRIVACY AND SECURITY RULE

§114A-2-1. General.

1.1. Scope. -- This legislative rule implements the privacy and security provisions of the All-Payer Claims Database Program found at W. Va. Code §33-4A-1 *et seq.* as administered by the Department of Health and Human Resources, the Offices of the Insurance Commissioner, and the West Virginia Health Care Authority.

1.2. Authority. -- W. Va. Code §§33-4A-4(b), (d), 33-4A-8(a) and (e).

1.3. Filing Date. – _____, 2012

1.4. Effective Date. – _____, 2012

§114A-2-2. Definitions.

As used in this legislative rule, all terms that are defined in section 1 of the Act have those same meanings which are in some cases further clarified in this section. Terms not defined in the Act have the following meanings unless the context expressly requires otherwise.

2.1. "Act" means the all-payer claims database act, W. Va. Code §33-4A-1 *et seq.*

2.2. "All-payer claims database" or "APCD" means the program authorized by this article that collects, retains, uses and discloses information concerning the claims and administrative expenses of health care payers.

2.2. "Chair" means the chairperson of the West Virginia Health Care Authority.

2.3. "Commissioner" means the West Virginia Insurance Commissioner.

2.4. "Data" means the data elements from enrollment and eligibility files, specified types of claims, and reference files for data elements not maintained in formats consistent with national coding standards.

2.5. "Health care payer" means any entity that pays or administers the payment of health insurance claims or medical claims under workers' compensation insurance to providers in this state, including workers' compensation insurers; accident and sickness insurers; nonprofit hospital service corporations, medical service corporations and dental service organizations; nonprofit health service corporations; prepaid limited health service organizations; health maintenance organizations; and government payers, including but not limited to Medicaid, Medicare and the public employees insurance agency; the term also includes any third-party administrator including any pharmacy benefit manager, that administers a fully-funded or self-funded plan: A "health insurance claim" does not include:

2.5.a. Any claim paid under an individual or group policy providing coverage only for accident, or disability income insurance or any combination thereof; coverage issued as a supplement to liability insurance; liability insurance, including general liability insurance and automobile liability; credit-only insurance; coverage for on-site medical clinics; other similar insurance coverage, which may be specified by rule, under which benefits for medical care are secondary or incidental to other insurance benefits; or,

2.5.b. Any of the following if provided under a separate policy, certificate, or contract of

insurance: Limited scope dental or vision benefits: benefits for long-term care, nursing home care, home health care, community-based care, or any combination thereof; coverage for only a specified disease or illness; or hospital indemnity or other fixed indemnity insurance.

2.5.c. "Health insurance claims" shall only include information from Medicare supplemental policies if the same information is obtained with respect to Medicare.

2.6. "MOU parties" means the secretary, commissioner and chair, collectively.

2.7. "Personal identifiers" means information relating to an individual member or insured that identifiers, or can be used to identify, locate or contact a particular individual member or insured, including but not limited to the individual's name, street address, social security number, e-mail address and telephone number.

2.8. "Secretary" means the Secretary of the West Virginia Department of Health and Human Services.

2.9. "Third-party administrator" has the same meaning ascribed to it in section two, article forty-six of this chapter.

§114A-2-3. Data Collection Privacy and Security Requirements.

3.1. All data transmitted by health care payers to the APCD shall be transmitted over a secure electronic communications network, provided by the commissioner or his or her vendor. Transmission to the commissioner, or his or her data collection vendor, shall constitute transmission to the APCD.

3.2. Transmission of the data from each health care payer to the APCD shall be in a secure manner that prevents unauthorized access and ensures confidentiality, integrity and availability. This data transmission shall be secured to the level required by the HIPAA Security and Privacy Rules, 45 CFR § 164.102 *et seq.* and shall be encrypted per NIST Special Publication 800-52, Guidelines for the Selection and use of Transport Layer Security Implementations, National Institute of Standards and Technology, June 2005, as amended or superseded.

§114A-2-4. Data Retention and Initial Use Privacy and Security Requirements.

4.1. All data retained by the APCD program shall be retained in a secure manner that prevents unauthorized access and ensures confidentiality, integrity and availability of all data transmitted to the APCD, at the levels required by the HIPAA Security and Privacy Rules, 45 CFR § 164.102 *et seq.* and shall be encrypted per NIST Special Publication 800-111, Guide to Storage Encryption Technologies for End User Devices, November 2007, as amended or superseded.

4.2. The MOU parties shall only use the data to assess the completeness and quality of health care payers' submissions in order to determine compliance with established data reporting requirements and standards. For purposes of this initial use, all personal identifiers shall remain encrypted and not visible to the MOU parties. Results of the completeness and quality assessments may be shared with the Advisory Group.

4.3. No additional uses and no disclosures contemplated by this program shall be made until such time as the MOU parties promulgate rules specifically delineating the same.

Joy Zirkle

From: Milam, Sallie [SMilam@hcawv.org]
Sent: Monday, April 25, 2011 4:39 PM
To: Joy Zirkle
Subject: RE: Legislation--Title 114A, Series 2-APCD Privacy and Security Rule-JDL markup--1104201615

Thank you.

Sallie Milam, JD, CIPP, CIPP/G
West Virginia's Chief Privacy Officer
WV Health Care Authority

From: Joy Zirkle [mailto:Joy.Zirkle@wvinsurance.gov]
Sent: Monday, April 25, 2011 3:44 PM
To: Milam, Sallie; Timothy Murphy
Cc: Debra Rayburn; Robin Hughes
Subject: RE: Legislation--Title 114A, Series 2-APCD Privacy and Security Rule-JDL markup--1104201615

I will make 2 copies each and have them ready.

Joy L. Zirkle
Offices of the Insurance Commissioner
Legal Division
P.O. Box 50540
Charleston, WV 25305-0540
Tel. (304) 558-0401
Fax (304) 558-1362

From: Timothy Murphy
Sent: Monday, April 25, 2011 3:17 PM
To: Joy Zirkle
Subject: FW: Legislation--Title 114A, Series 2-APCD Privacy and Security Rule-JDL markup--1104201615

Timothy Murphy
Associate Counsel
West Virginia Offices of the Insurance Commission
Legal Division
P. O. Box 50540
Charleston, WV 25305-0540
phone 304-558-6279 ext. 1210
fax 304-558-1362

NIST
National Institute of
Standards and Technology
U.S. Department of Commerce

Special Publication 800-111

Guide to Storage Encryption Technologies for End User Devices

Recommendations of the National Institute of Standards and Technology

Karen Scarfone
Murugiah Souppaya
Matt Sexton

NIST Special Publication 800-111

Guide to Storage Encryption Technologies for End User Devices

Recommendations of the National Institute of Standards and Technology

Karen Scarfone
Murugiah Souppaya
Matt Sexton

C O M P U T E R S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

November 2007

U.S. Department of Commerce
Carlos M. Gutierrez, Secretary
National Institute of Standards and Technology
James M. Turner, Acting Director

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive unclassified information in Federal computer systems. This Special Publication 800-series reports on ITL's research, guidance, and outreach efforts in computer security and its collaborative activities with industry, government, and academic organizations.

National Institute of Standards and Technology Special Publication 800-111
Natl. Inst. Stand. Technol. Spec. Publ. 800-111, 40 pages (Nov. 2007)

Acknowledgements

The authors, Karen Scarfone and Murugiah Souppaya of the National Institute of Standards and Technology (NIST), and Matt Sexton of Booz Allen Hamilton, wish to thank their colleagues who reviewed drafts of this document and contributed to its technical content. In particular, their appreciation goes to Tim Grance, Bill Burr, and Tim Polk of NIST, and Derrick Dicoi, Angela Orebaugh, Manuel Villar, Mike Zeberlein, and Mike Zirkle of Booz Allen Hamilton, for their keen and insightful assistance throughout the development of the document. The authors also appreciate the efforts of those individuals, agencies, and other organizations that contributed input during the public comment period.

Table of Contents

List of Tables

Executive Summary

In today's computing environment, there are many threats to the confidentiality of information stored on end user devices, such as personal computers, consumer devices (e.g., personal digital assistant, smart phone), and removable storage media (e.g., universal serial bus [USB] flash drive, memory card, external hard drive, writeable CD or DVD). Some threats are unintentional, such as human error, while others are intentional. Intentional threats are posed by people with many different motivations, including causing mischief and disruption and committing identity theft and other fraud. A common threat against end user devices is device loss or theft. Someone with physical access to a device has many options for attempting to view or copy the information stored on the device. Another concern is insider attacks, such as an employee attempting to access sensitive information stored on another employee's device. Malware, another common threat, can give attackers unauthorized access to a device, transfer information from the device to an attacker's system, and perform other actions that jeopardize the confidentiality of the information on a device.

Many threats against end user devices could cause information stored on the devices to be accessed by unauthorized parties. To prevent such disclosures of information, particularly of personally identifiable information (PII) and other sensitive data, the information needs to be secured. Securing other components of end user devices, such as operating systems, is also necessary, but in many cases additional measures are needed to secure the stored information. This publication explains the basics of storage security, which is the process of allowing only authorized parties to access and use stored information. The primary security controls for restricting access to sensitive information stored on end user devices are encryption and authentication. Encryption can be applied granularly, such as to an individual file containing sensitive information, or broadly, such as encrypting all stored data. The appropriate encryption solution for a particular situation depends primarily upon the type of storage, the amount of information that needs to be protected, the environments where the storage will be located, and the threats that need to be mitigated. This publication describes three types of solutions—full disk encryption, volume and virtual disk encryption, and file/folder encryption—and makes recommendations for implementing and using each type. This publication also includes several use case examples, which illustrate that there are multiple ways to meet most storage encryption needs. When selecting a solution type, organizations should consider the range of solutions that meet the organization's security requirements, and not just the solution type that is most commonly used.

Implementing the following recommendations should facilitate more efficient and effective storage encryption solution design, implementation, and management for Federal departments and agencies.

When selecting a storage encryption technology, organizations should consider solutions that use existing system features (such as operating system features) and infrastructure.

There are many factors for organizations to consider when selecting storage encryption solutions, such as the platforms they support, the data they protect, and the threats they mitigate. Some solutions involve deploying various servers and installing software on the devices to be protected, while other solutions can use existing servers, as well as software built into the devices to be protected, such as Federal Information Processing Standard (FIPS) approved encryption features built into the devices' operating systems. Generally, the more extensive the changes are to the infrastructure and devices, the more likely it is that the storage encryption solution will cause a loss of functionality or other problems with the devices. When evaluating solutions, organizations should compare the loss of functionality with the gain in security capabilities and decide if the tradeoff is acceptable. Solutions that require extensive changes to the infrastructure and end user devices should generally be used only when other solutions cannot meet the organization's needs.

Organizations should use centralized management for all deployments of storage encryption except for standalone deployments and very small-scale deployments.

Centralized management is recommended for most storage encryption deployments because of its effectiveness and efficiency for policy verification and enforcement, key management, authenticator management, data recovery, and other management tasks. Centralized management can also automate deployment and configuration of storage encryption software to end user devices, distribution and installation of updates, collection and review of logs, and recovery of information from local failures.

Organizations should ensure that all cryptographic keys used in a storage encryption solution are secured and managed properly to support the security of the solution.

Storage encryption technologies use one or more cryptographic keys to encrypt and decrypt the data that they protect. If a key is lost or damaged, it may not be possible to recover the encrypted data from the computer. Therefore, organizations should perform extensive planning of key management processes, procedures, and technologies before implementing storage encryption technologies. This planning should include all aspects of key management, including key generation, use, storage, recovery, and destruction. Organizations should carefully consider how key management practices can support the recovery of encrypted data if a key is inadvertently destroyed or otherwise becomes unavailable. Organizations planning on encrypting removable media also need to consider how changing keys will affect access to encrypted storage on removable media and develop feasible solutions, such as retaining the previous keys in case they are needed.

Organizations should select appropriate user authenticators for storage encryption solutions.

Storage encryption solutions require users to authenticate successfully before accessing the information that has been encrypted. Common authentication mechanisms are passwords, personal identification numbers, cryptographic tokens, biometrics, and smart cards. Organizations should consider leveraging existing enterprise authentication solutions (e.g., Active Directory, public key infrastructure [PKI]) instead of adding another authenticator for users. Generally, this is acceptable if two-factor authentication is being used. Using the same single-factor authenticator for multiple purposes, such as operating system (OS) authentication and storage encryption authentication, significantly weakens the protection that authentication provides. For example, an attacker who learns a single password could gain full access to the device's information. Organizations should carefully consider the security implications of using the same single-factor authenticator for multiple purposes. In particular, organizations should not use email passwords and other passwords sometimes transmitted in plaintext as single-factor authenticators for storage encryption.

Organizations should implement measures that support and complement storage encryption implementations for end user devices.

Storage encryption by itself cannot provide adequate security for stored information; additional security controls are needed. Organizations should select and deploy the necessary controls based on FIPS 199's categories for the potential impact of a security breach involving a particular system and NIST Special Publication 800-53's recommendations for minimum management, operational, and technical security controls. Examples of supporting controls are as follows:

- Revising organizational policies as needed to incorporate appropriate usage of the storage encryption solution.

- Securing and maintaining end user devices properly, which should reduce the risk of compromise or misuse. This includes securing device operating systems, applications, and communications, and physically securing devices.

- Making users aware of their responsibilities for storage encryption, such as encrypting sensitive files, physically protecting mobile devices and removable media, and promptly reporting loss or theft of devices and media.

1. Introduction

1.1 Authority

The National Institute of Standards and Technology (NIST) developed this document in furtherance of its statutory responsibilities under the Federal Information Security Management Act (FISMA) of 2002, Public Law 107-347.

NIST is responsible for developing standards and guidelines, including minimum requirements, for providing adequate information security for all agency operations and assets; but such standards and guidelines shall not apply to national security systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), "Securing Agency Information Systems," as analyzed in A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided in A-130, Appendix III.

This guideline has been prepared for use by Federal agencies. It may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright, though attribution is desired.

Nothing in this document should be taken to contradict standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority, nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official.

1.2 Purpose and Scope

The purpose of this document is to assist organizations in understanding storage encryption technologies for end user devices and in planning, implementing, and maintaining storage encryption solutions. The types of end user devices addressed in this document are personal computers (desktops and laptops), consumer devices (e.g., personal digital assistants, smart phones), and removable storage media (e.g., USB flash drives, memory cards, external hard drives, writeable CDs and DVDs). This document provides practical, real-world guidance for three classes of storage encryption techniques: full disk encryption, volume and virtual disk encryption, and file/folder encryption. It also discusses important security elements of a storage encryption deployment, including cryptographic key management and authentication. It only discusses the encryption of data at rest (storage), and does not address the encryption of data in motion (transmission).

1.3 Audience

This document has been created for information security program managers and staff, system administrators, and others who are responsible for selecting, deploying, managing, and maintaining storage encryption technologies for end user devices. This document does not assume that the reader has previous experience with any storage encryption technologies, but it does assume that the reader has experience with information security.

1.4 Document Structure

The remainder of this document is organized into three major sections.

- Section 2 provides an overview of the basic concepts of storage encryption for end user devices.

- Section 3 describes the most commonly used categories of storage encryption technologies for end user devices, and explains the types of protection they provide.
- Section 4 discusses the process of planning and implementing storage encryption technologies for end user devices. It includes a detailed discussion of the importance of cryptography and authentication to a storage encryption solution.

The document also contains several appendices with supporting material.

- Appendix A describes alternatives to encrypting storage on end user devices.
- Appendices B and C contain a glossary and acronym list, respectively.
- Appendix D lists online tools and resources that may be useful references for gaining a better understanding of storage encryption for end user devices.

2. Storage Security Overview

An *end user device* is a personal computer (desktop or laptop), consumer device (e.g., personal digital assistant [PDA], smart phone), or removable storage media (e.g., USB flash drive, memory card, external hard drive, writeable CD or DVD) that can store information.[1] *Storage security* is the process of allowing only authorized parties to access and use stored information. This section introduces the basic concepts of storage security for end user devices.[2]

2.1 File Storage Basics

A *file* is a collection of information logically grouped into a single entity and referenced by a unique name, such as a *filename*. On end user devices, there are typically two types of files: data files, such as text documents, spreadsheets, images, and videos, and system files, such as operating system and application binaries and libraries. A *filesystem* defines the way that files are named, stored, organized, and accessed. *Directories*, also known as *folders*, are organizational structures used by filesystems to group files. Another feature of a filesystem is *metadata*, which is data about data—in the context of a filesystem, information regarding the files and folders themselves, such as file and folder names, creation dates and times, and sizes.

Filesystems are designed to store folders, system and data files, and metadata on storage media. However, storage media may also hold *residual data*, which is data from deleted files (including earlier versions of existing files and temporary files). Residual data can often be recovered from an end user device through forensic analysis. The following items describe common forms of residual data:

- **Unused File Allocation Units.** Filesystems store files in chunks known as file allocation units. *Unused file allocation units* are the units within a partition that are not currently being used by the filesystem. When a file is deleted, it is typically not erased from the media; instead, the information in the directory's data structure that points to the location of the file is marked as deleted. This means that the file is still stored on the media but is no longer enumerated by the operating system (OS). The OS considers this to be unused space and can overwrite any portion of or the entire deleted file at any time.
- **Slack Space.** Even if a file requires less space than the file allocation unit size, an entire file allocation unit is still reserved for the file. For example, if the file allocation unit size is 32 kilobytes (KB) and a file is only 7 KB, the entire 32 KB is still allocated to the file, but only 7 KB is used, resulting in 25 KB of unused space. This unused space is referred to as *file slack space,* and it may hold residual data such as portions of deleted files.
- **Free Space.** *Free space* is the area on media that is not currently allocated to a partition. This often includes space on the media where files may have resided at one point but have since been deleted. The free space may still contain pieces of data.

Before media can be used to store files, the media must usually be partitioned and formatted into logical volumes. *Partitioning* is the act of logically dividing a media into portions that function as separate units. A *logical volume* is a partition or a collection of partitions acting as a single entity that has been formatted

[1] This publication only addresses technologies for encrypting files stored on end user devices. Information on storage security for servers, storage area networks, enterprise backup tapes, and other devices is outside the scope of this publication.

[2] Storage security is only one component of data security, which includes network, host, and application security, and also addresses how data may be used after it is accessed. All elements of data security other than storage security, such as encrypting data in motion (e.g., network communications), are outside the scope of this publication.

with a filesystem. Some media types can contain only one partition (and consequently, one logical volume), while others can contain multiple partitions.

2.2 The Need for Storage Security

In today's computing environment, there are many threats to the confidentiality of information stored on end user devices. Some threats are unintentional, such as human error, while others are intentional. Intentional threats are posed by people with many different motivations, including causing mischief and disruption and committing identity theft and other fraud. One of the most common threats is *malware*, also known as *malicious code*, which refers to a program that is inserted into a system, usually covertly, with the intent of compromising the confidentiality, integrity, or availability of the victim's data, applications, or OS. Types of malware threats include viruses, worms, malicious mobile code, Trojan horses, rootkits, and spyware. Malware can give attackers unauthorized access to a device, transfer information from the device to an attacker's system, and perform other actions that jeopardize the confidentiality of the information on a device. Another common threat against end user devices is device loss or theft. Someone with physical access to a device has many options for attempting to view the information stored on the device. This is also a concern for insider attacks, such as an employee attempting to access sensitive information stored on another employee's device. Another form of insider attack is a user attempting to access another user's files on a device that the two users share.

Many threats against end user devices could cause information stored on the devices to be accessed by unauthorized parties. To prevent such disclosures of information, particularly of personally identifiable information (PII)[3] and other sensitive data, the information needs to be secured. Securing other components of end user devices, such as OSs, is also necessary, but in many cases additional measures are needed to secure the stored information. For example, without these additional measures, an attacker that steals a device could use forensic tools and techniques to recover information directly from the storage media, circumventing the protections applied by the device's OS.

A number of laws and regulations compel organizations to ensure that sensitive information is protected appropriately. The following is a list of key regulations, standards, and guidelines that help define organizations' needs for storage security:[4]

- **Federal Information Security Management Act of 2002 (FISMA).** FISMA emphasizes the need for each Federal agency to develop, document, and implement an organization-wide program to provide information security for the information systems that support its operations and assets. NIST Special Publication (SP) 800-53, *Recommended Security Controls for Federal Information Systems*, was developed in support of FISMA.[5] NIST SP 800-53 is the primary source of recommended security controls for Federal agencies. It describes several controls related to storage security, such as controlling access through encryption of stored information, restricting access to mobile computing devices and information system media, and storing media in physically secure locations.

---

[3] OMB Memorandum 06-19, "Reporting Incidents Involving Personally Identifiable Information and Incorporating the Cost for Security in Agency Information Technology Investments", defines PII as "any information about an individual maintained by an agency, including, but not limited to, education, financial transactions, medical history, and criminal or employment history and information which can be used to distinguish or trace an individual's identity, such as their name, social security number, date and place of birth, mother's maiden name, biometric records, etc., including any other personal information which is linked or linkable to an individual." The full text of the memorandum is available at http://www.whitehouse.gov/omb/memoranda/fy2006/m-06-19.pdf.

[4] It is outside this publication's scope to explain which types of information organizations need to protect and how each type should be protected. NIST SP 800-60, *Guide for Mapping Types of Information and Information Systems to Security Categories*, discusses the identification of common types of information. It is at http://csrc.nist.gov/publications/nistpubs/.

[5] Copies of FISMA and NIST SP 800-53 are available at http://csrc.nist.gov/sec-cert/ca-library.html.

- **OMB Memorandum M-06-16.** OMB has issued a memorandum directly related to storage security. OMB M-06-16 addresses the protection of agency information that is either "accessed remotely or physically transported outside of the agency's secured, physical perimeter". It specifically requires that agencies encrypt all data stored on mobile computing devices, such as laptops and PDAs, unless the data has been determined by the designated agency official to be non-sensitive.[6] Similar requirements are also included in OMB Memorandum M-07-16.[7]

- **Privacy Act of 1974.** The Privacy Act regulates the collection, use, maintenance, and dissemination of personal information about U.S. citizens or aliens lawfully admitted for permanent residence. It applies to records maintained by agencies in the executive branch of the government.

- **Gramm-Leach-Bliley Act (GLBA).** GLBA requires financial institutions to protect their customers' information against security threats. This includes ensuring "the security and confidentiality of customer records and information" and protecting "against unauthorized access to or use of such records or information".[8]

- **Health Insurance Portability and Accountability Act of 1996 (HIPAA).** HIPAA includes security standards for certain health information. NIST SP 800-66, *An Introductory Resource Guide for Implementing the Health Insurance Portability and Accountability Act (HIPAA) Security Rule*, lists HIPAA-related storage security needs.[9] For example, Section 4.14 of NIST SP 800-66 describes the need to encrypt and decrypt electronic protected health information (EPHI).

2.3 Security Controls for Storage

The primary security controls for restricting access to sensitive information stored on end user devices are encryption and authentication. Encryption can be applied granularly, such as to an individual file containing sensitive information, or broadly, such as encrypting all stored data. The appropriate encryption solution for a particular situation depends primarily upon the type of storage, the amount of information that needs to be protected, the environments where the storage will be located, and the threats that need to be mitigated. Section 3 discusses the most commonly used options in detail; Appendix A briefly discusses some additional options. Storage encryption solutions require users to authenticate successfully before accessing the information that has been encrypted. Common authentication mechanisms are passwords, personal identification numbers (PIN), cryptographic tokens, biometrics, and smart cards. The combination of encryption and authentication helps control access to the stored information.

Organizations also need to consider the security of backups of stored information. Some organizations permit users to back up their local files to a centralized system, while other organizations recommend that their users perform local backups (e.g., burning CDs, external USB storage media). In the latter case, organizations should ensure that the backups will be secured at least as well as the original source. This could be done with similar controls, such as encrypting the backups, or with different types of controls, such as storing backup tapes in a physically secured room within the organization's facilities.

Organizations should also implement other measures that support and complement storage encryption implementations. These measures help to ensure that storage encryption is implemented in an

---

6 The memorandum is available at http://www.whitehouse.gov/OMB/memoranda/fy2006/m06-16.pdf.

7 M-07-16 is available at http://www.whitehouse.gov/omb/memoranda/fy2007/m07-16.pdf.

8 More information on GLBA is available at http://www.ftc.gov/privacy/privacyinitiatives/glbact.html. A copy of GLBA can be downloaded from http://www.ftc.gov/privacy/privacyinitiatives/financial_rule_lr.html.

9 HIPAA is available at http://www.hhs.gov/ocr/hipaa/, and NIST SP 800-66 is available at http://csrc.nist.gov/publications/nistpubs/.

environment with the management, operational, and technical controls necessary to provide adequate security for the storage encryption implementation. Examples of supporting measures are as follows:

- Revise organizational policies as needed to incorporate appropriate usage of the storage encryption solution. Policies should provide the foundation for the planning and implementation of storage encryption.

- Ensure that end user devices are secured and maintained properly, which should reduce the risk of compromise or misuse. This includes securing device OSs, applications, and communications (e.g., encrypting wired and wireless network traffic) and physically securing devices, such as requiring that laptops be secured using cable locks when in hotels, conferences, and other locations where third parties could easily gain physical access to the devices. Physical security for devices is also an important consideration in home environments, such as preventing others within the house from using an organization-issued device by keeping the device in a locked desk or room.

- Make users aware of their responsibilities for storage encryption, such as encrypting sensitive files, physically protecting mobile devices and removable media, and promptly reporting loss or theft of devices and media.

Organizations should select and deploy the necessary security controls based on existing guidelines. Federal Information Processing Standards (FIPS) 199 establishes three security categories—low, moderate, and high—based on the potential impact of a security breach involving a particular system.[10] NIST SP 800-53 provides recommendations for minimum management, operational, and technical security controls for information systems based on the FIPS 199 impact categories.[11] The recommendations in NIST SP 800-53 should be helpful to organizations in identifying controls that are needed to protect end user devices, which should be used in addition to the specific recommendations for storage encryption listed in this document.[12]

---

10 FIPS 199, *Standards for Security Categorization of Federal Information and Information Systems*, is available at http://csrc.nist.gov/publications/fips/.

11 NIST SP 800-53 Revision 1, *Recommended Security Controls for Federal Information Systems*, is available at http://csrc.nist.gov/publications/nistpubs/.

12 In addition to securing the wireless networks, the wireless devices using the networks also need to be secured; however, an explanation of securing laptops, PDAs, and other wireless devices is outside the scope of this guide.

3. Storage Encryption Technologies

There are many technologies available for encrypting data stored on end user devices. This section describes the most commonly used technologies, discusses the protections provided by each type, and explains how these technologies are typically managed.

3.1 Common Types of Storage Encryption Technologies

This section provides a high-level overview of the most commonly used options for encrypting stored information: full disk encryption, volume and virtual disk encryption, and file/folder encryption.[13] It briefly defines each option and explains at a high level how it works.

3.1.1 Full Disk Encryption

Full disk encryption (FDE), also known as whole disk encryption, is the process of encrypting all the data on the hard drive used to boot a computer, including the computer's OS, and permitting access to the data only after successful authentication to the FDE product. Most FDE products are software-based, so this section focuses on explaining the capabilities and characteristics of software-based FDE solutions. Hardware-based solutions are discussed briefly at the end of this section.

FDE software works by redirecting a computer's *master boot record (MBR)*, which is a reserved sector on bootable media that determines which software (e.g., OS, utility) will be executed when the computer boots from the media. Before FDE software is installed onto a computer, the MBR usually points to the computer's primary OS. When FDE software is being used, the computer's MBR is redirected to a special pre-boot environment (PBE) that controls access to the computer.[14] This redirection is depicted in Figure 3-1. The PBE prompts the user to authenticate successfully, such as entering a user ID and password, before decrypting and booting the OS. This is known as *pre-boot authentication (PBA)*.[15] Most FDE products support the use of both network-based authentication (e.g., Active Directory, PKI) and local authentication sources (e.g., locally stored, locally cached from network source) for PBA.

Once successful PBA occurs, the FDE software decrypts the boot sector for the OS, as depicted by the second arrow in Figure 3-1, and the boot loader in the boot sector starts to load the OS. As it loads, the FDE software decrypts the OS files (which are stored in the system volume) as needed, indicated in Figure 3-1 by the third arrow. Once the OS has finished booting, the user provides OS authentication and uses the computer normally. When the user needs to open encrypted files, save new files, or perform other operations involving the hard drive, the FDE software transparently decrypts and encrypts the necessary sectors[16] of the hard drive as needed.[17] This may marginally increase the time needed to open or save files, but the delay generally should only be noticeable for particularly large files. On an FDE-protected computer, users will typically notice a delay of at least a few seconds when booting the

---

[13] Information stored on end user devices can be encrypted in many ways. For example, an application that accesses sensitive information could be responsible for encrypting that information. Applications such as backup programs might also offer encryption options. Another method for protecting files is digital rights management (DRM) software.

[14] Some FDE implementations verify the integrity of the boot components, including the MBR, before proceeding. Figure 3-1 depicts the PBE as being stored in the space between the MBR and the boot sector; although many software-based FDE products store the PBE there, this is not required, and some products store the PBE elsewhere.

[15] Most software-based FDE products use PBA. Other products require the user to authenticate after the OS has booted, which provides weaker protection than PBA. This publication assumes that FDE implementations are configured to require PBA. PBA can be performed with stronger authenticators than user ID and password; see Section 4.2 for additional information.

[16] A *sector* is the smallest logical component of a hard drive that can be read or written. Many hard drives have 512-byte sectors. Even if a single byte of data needs to be accessed, the hard drive will read the entire sector containing that data.

[17] Figure 3-1 does not show data volumes and other volumes that might be on a hard drive; such volumes would also be encrypted by an FDE solution.

computer or shutting it down. Delays may also occur when using hibernation[18] features, because the FDE software has to encrypt and decrypt the large hibernation file (which includes a copy of the computer's memory) that is stored on the hard drive. The length of delays is dependent on the size of memory, the hard drive's size and speed, and other factors.

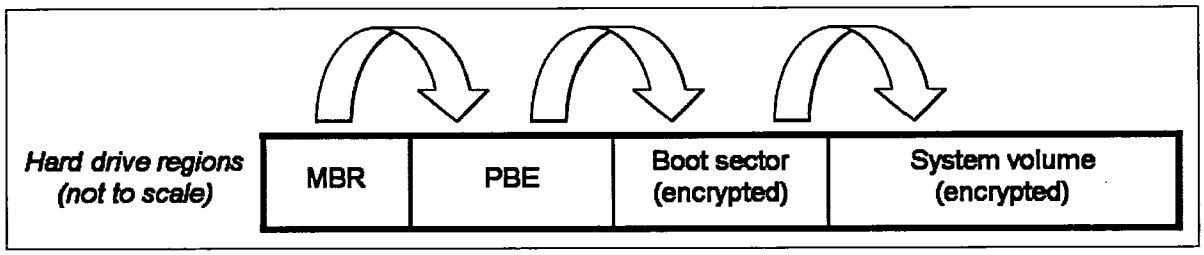

Figure 3-1. Boot Sequence for Full Disk Encryption Software

Because FDE alters how a computer boots, it can cause operational problems. For example, modifying the MBR can prevent computers with dual-boot configurations from functioning properly,[19] and storing the PBE in the space between the MBR and the boot sector can cause conflicts with other software, such as disk-level software tools, that also store code in that space. FDE-protected devices may also have problems with asset management tools and the use of wake-on-LAN.[20]

FDE software is most commonly used on desktop and laptop computers. The requirement for pre-boot authentication means that users have to be able to authenticate using the most fundamental components of a device, such as a standard keyboard—because the OS is not loaded, OS-level drivers are unavailable. For example, a PDA or smart phone could not display a keyboard on the screen for entering a password because that is an OS-level capability.

As mentioned at the beginning of the section, FDE can also be built into a hard drive disk controller.[21] Hardware and software-based FDE offer similar capabilities through different mechanisms. When a user tries to boot a device protected with hardware-based FDE, the hard drive prompts the user to authenticate before it allows an OS to load. The FDE capability is built into the hardware in such a way that it cannot be disabled or removed from the drive. The encryption code and authenticators, such as passwords and cryptographic keys, are stored securely on the hard drive. Because the decryption and encryption is performed by the hard drive itself, with no OS participation, typically there is very little performance impact.

---

[18] Hibernation refers to saving the state of the computer, including the contents of memory, and placing the computer in a low power-usage mode that uses just enough power to maintain its state. Depending on the OS, hibernation mode may also be called sleep, standby, or suspend mode; however, some of these terms do not have universally accepted definitions, and some OSs have features with these names that do not actually write memory out to a file. Modes that do not write memory to a file should not be used with FDE software because the FDE software will not protect the data in these modes.

[19] Dual-boot computers usually modify the MBR to enable users to select which OS will be booted. The details of this depend on the OSs and any utilities used to manage the disk partitions. Using FDE with a dual-boot computer increases the complexity of the configuration and the likelihood for errors that cause loss of data or loss of availability.

[20] The problem with wake-on-LAN technologies is the requirement to perform PBA before booting the system. Some FDE products can be configured to skip PBA when wake-on-LAN is used, either every time or for a certain number of reboots. Access to the OS logon can be suppressed to somewhat compensate for the lack of PBA. However, skipping PBA during wake-on-LAN still introduces additional risk, because an attacker that takes a device that is configured this way could easily set up wake-on-LAN services, circumvent PBA, and use forensic tools to gain access to the device's information. Organizations should carefully consider these risks before using wake-on-LAN for FDE-protected devices.

[21] As of mid-2007, few hardware-based FDE products were yet available, but vendors had announced that several additional products would be available in the coming months.

A major difference between software and hardware-based FDE is that software-based FDE can be centrally managed, but hardware-based FDE can usually only be managed locally. This makes key management and recovery actions considerably more resource-intensive and cumbersome for hardware-based FDE than software-based. Another major difference is that because hardware-based FDE does all cryptographic processing within the hard drive's hardware, it does not need to place its cryptographic keys into the computer's memory, which could potentially expose the keys to malware and other threats. A third significant difference is that hardware-based FDE typically does not alter the MBR, so hardware-based FDE does not cause conflicts with software that modifies the MBR (e.g., dual-boot configurations).

3.1.2 Virtual Disk Encryption and Volume Encryption

Virtual disk encryption is the process of encrypting a file called a *container*, which can hold many files and folders, and permitting access to the data within the container only after proper authentication is provided, at which point the container is typically mounted as a virtual disk. Virtual disk encryption is used on all types of end user device storage. The container is a single file that resides within a logical volume. Examples of volumes are boot, system, and data volumes on a personal computer, and a USB flash drive formatted with a single filesystem. *Volume encryption* is the process of encrypting an entire logical volume and permitting access to the data on the volume only after proper authentication is provided. Volume encryption is most often performed on hard drive data volumes and volume-based removable media, such as USB flash drives and external hard drives. Volume encryption of boot and system volumes is essentially a special form of FDE, and it is not discussed in this section; see the FDE material in Section 3.1.1 for additional information.

At a high level, volume and virtual disk encryption are performed similarly. Software running on the OS used to access the volume or container handles all attempts to read to or write from the protected volume or container.[22] Once the OS has been loaded, if the user needs to use the encrypted volume or container, it will be mounted after the user has provided the required authentication. The software will then automatically decrypt and encrypt the appropriate sectors as needed. This increases the time needed to open or save files, but the delay generally should be noticeable for only particularly large files. There may also be slight delays associated with mounting and unmounting an encrypted volume or container.

The key difference between volume and virtual disk encryption is that containers are portable and volumes are not—a container can be copied from one medium to another, with encryption intact. This allows containers to be burned to CDs and DVDs and to be used on other media that are not volume-based. Virtual disk encryption also makes it trivial to back up sensitive data; the container is simply copied to the backup server or media. Another advantage of virtual disk encryption over volume encryption is that virtual disk encryption can be used in situations where volume-based removable media needs to have both protected and unprotected storage; the volume can be left unprotected and a container placed onto the volume for the sensitive information.

Some virtual disk encryption products further support mobility by offering features that can place executables on the medium holding a container. The medium can then be moved to another computer and the executables run, through methods such as installing drivers onto the computer or running an authentication and decryption utility.[23] The protected contents of the medium can then be accessed by a user after providing the requested authentication.

The responsibilities of the users of virtual disk and volume encryption solutions vary, primarily depending on the devices' access control. For example, if a laptop's OS is configured so that a user can

[22] Some products install kernel mode drivers to perform volume and virtual disk encryption. Other products, especially those specifically designed for removable media, either contain their own resident OSs or provide software applications.

[23] This only speaks to the portability of the logical entity on the media—the media itself might be physically portable.

only write files to an encrypted container or volume, then the user does not need to take steps to ensure that files are saved to the appropriate location. However, if the OS is not configured this way, permitting users to save files to various locations, or if the encrypted device is removable media that is not protected through OS access control features, then users will be responsible for ensuring that they save files in the appropriate location. In this case, if users fail to follow the necessary procedures, then some files that should be protected may not be.

3.1.3 File/Folder Encryption

File encryption is the process of encrypting individual files on a storage medium and permitting access to the encrypted data only after proper authentication is provided. *Folder encryption* is very similar to file encryption, only it addresses individual folders instead of files. Some OSs offer built-in file and/or folder encryption capabilities,[24] and many third-party programs are also available. Although folder encryption and virtual disk encryption sound similar—both a folder and a container are intended to contain and protect multiple files—there is a difference. A container is a single opaque file, meaning that no one can see what files or folders are inside the container until the container is decrypted. File/folder encryption is transparent, meaning that anyone with access to the filesystem can view the names and possibly other metadata for the encrypted files and folders, including files and folders within encrypted folders, if they are not protected through OS access control features. File/folder encryption is used on all types of storage for end user devices.

File/folder encryption can be implemented in many ways, including through drivers, services, and applications. When a user attempts to open an encrypted file (either encrypted by itself or located in an encrypted folder), the software requires the user to first authenticate successfully. Once that has been done, the software will automatically decrypt the chosen file. Because it decrypts a single file at a time, the performance impact of file/folder encryption should be minimal. File/folder encryption is most commonly used on user data files, such as word processing documents and spreadsheets. File/folder encryption solutions can sometimes encrypt swap files, but typically not OS executables and hibernation files.

Many file/folder encryption products offer several options for selecting which files and folders should be encrypted and defining the user's role in using the solution—manually enabling encryption for each new file or folder that needs protection, remembering to store files and folders in the proper locations, or doing nothing differently because the files and folders are encrypted automatically. Common options include:

- Relying on the user to specifically designate the files and folders
- Automatically encrypting the contents of administrator-designated folders
- Automatically encrypting certain types of files, such as those with a particular file extension
- Automatically encrypting all files written to by particular applications
- Automatically encrypting all data files for particular users.

There are also various applications, such as file compression utilities and office productivity suites, that offer limited file/folder encryption capabilities. Such applications are usually completely dependent on the user to ensure that the necessary files are encrypted, and these applications are often not centrally managed, which can complicate key management and other aspects of managing the use of the

[24] For example, New Technology File System (NTFS) supports file and folder encryption using the Encrypting File System (EFS).

applications' file/folder encryption features. Appendix A presents additional examples of applications that can encrypt the information that they store.

3.2 Protection Provided by Storage Encryption Technologies

The following explains the types of protection each storage encryption technology can and cannot provide.

- **Full Disk Encryption.** For a computer that is not booted, all the information encrypted by FDE is protected, assuming that pre-boot authentication is required. When the device is booted, then FDE provides no protection; once the OS is loaded, the OS becomes fully responsible for protecting the unencrypted information. The exception to this is when the device is in a hibernation mode; most FDE products can encrypt the hibernation file.

- **Virtual Disk and Volume Encryption.** When virtual disk encryption is employed, the contents of containers are protected until the user is authenticated for the containers. If single sign-on is being used for authentication to the solution, this usually means that the containers are protected until the user logs onto the device. If single sign-on is not being used, then protection is typically provided until the user explicitly authenticates to a container. Virtual disk encryption does not provide any protection for data outside the container, including swap and hibernation files that could contain the contents of unencrypted files that were being held in memory. Volume encryption provides the same protection as virtual disk encryption, but for a volume instead of a container.

- **File/Folder Encryption.** File/folder encryption protects the contents of encrypted files (including files in encrypted folders) until the user is authenticated for the files or folders. If single sign-on is being used, this usually means that the files are only protected until the user logs onto the device. If single sign-on is not being used, then protection is typically provided until the user explicitly authenticates to a file or folder. File/folder encryption does not provide any protection for data outside the protected files or folders, including swap and hibernation files that could contain the contents of unencrypted files that were being held in memory. File/folder encryption software also cannot protect the confidentiality of filenames and other file metadata, which itself could provide valuable information to attackers (for examples, files that are named by Social Security number).

In many cases, especially for FDE and volume encryption, these products do not provide any protection for files copied or moved from the encrypted storage to another location (either local or on the network), because they automatically decrypt the files as part of the copy or move process.[25] The target location is responsible for protecting the files, and no protection is provided in transit from the source to the target. However, some storage encryption technologies allow protection to be retained if desired. Most virtual disk encryption products allow an entire container to be transferred, including the container's protection, but individual files or folders copied or moved from a container will not be protected. Some file/folder encryption products allow files or folders to retain their protection when they are copied or moved, in some cases only within a single filesystem, and in other cases both within a single filesystem and to other filesystems.

The main threat that all these types of technology mitigate is unauthorized access to information on a lost or stolen device. Virtual disk/volume encryption and file/folder encryption technologies can also mitigate some OS and application layer threats to protected information involving malware, remote access to the protected information, and other methods that depend on the OS being booted, until the user successfully

[25] Some products display a warning message or prompt the user to confirm the action before decrypting and copying or moving the files.

authenticates to the encryption solution. Once this authentication occurs, then any process being run on the device (such as malware) with access to the user's files can get the decrypted information. Because the files are only protected until successful authentication occurs, it may be beneficial to use a solution that is configured to encrypt only the necessary files (e.g., using file/folder encryption to encrypt 10 sensitive files instead of using volume encryption to encrypt 10 sensitive files and 1000 non-sensitive files). The more files that are protected, the sooner the user is likely to authenticate to the storage encryption solution, which increases the window of exposure for the decrypted files.

Some products also permit storage to be encrypted either for a single user or for multiple users of a device. If encrypted for a single user, the confidentiality of that user's encrypted storage is protected from other users of the device, including (in most cases) the device's administrators. Encrypting for multiple users allows sensitive data to be shared by those users, while still protecting it from other users of the device. This provides some protection against insider threats.

In some cases, multiple types of technology can be used concurrently to protect against different classes of threats; for example, FDE could be used to protect all data on a device from device loss or theft, and volume, virtual disk, or file/folder encryption could be used to provide additional protection for a subset of data that is more sensitive than the rest of the data.[26]

When thinking about threats, organizations should be aware that after storage encryption technology has been implemented, there may be residual data on the device that remains unprotected. For example, when a file is encrypted using file/folder encryption and the original file is deleted, the remnants of the original plaintext file might still be present on the storage media. Another example is FDE and volume encryption products that encrypt only the disk sectors that contain current files, not disk sectors that only contain deleted files or other remnants of data. These remnants may be recoverable using forensic tools by an attacker who gets physical access to the computer, without having to provide any authentication. Organizations should take into account threats against both the files and remnants of the files.

Organizations should be aware that if an end user device is compromised at any time, any storage encryption technologies on it may become partially or wholly ineffective. For example, when the device is in use and the user has been authenticated to the storage encryption solution, malware could access decrypted files and transfer copies of them to external hosts or extract sensitive information from them. Other examples are an attacker disabling or reconfiguring storage encryption, malware installing a keylogger that captures passwords used for storage encryption authentication, or malware acquiring a copy of a storage encryption key from the device's memory (for software-based storage encryption solutions).

Organizations should also be aware that they should not rely on storage encryption technologies to protect data without regularly maintaining the encryption solution. For example, if an attacker acquires a lost, stolen, or retired device protected by storage encryption technology, and a vulnerability in the storage encryption technology is discovered in the future, the attacker may be able to exploit it to access the protected data.

3.3 Comparison of Storage Encryption Technologies

Table 3-1 lists several characteristics of storage encryption technologies as a means for comparing the types of technologies described in this publication.

[26] When multiple encryption methods are used simultaneously, the cryptographic keys used by the encryption methods are usually different.

Table 3-1. Characteristics of Storage Encryption Technologies

Characteristic	Full Disk Encryption	Volume Encryption	Virtual Disk Encryption	File/Folder Encryption
Typical platforms supported	Desktop and laptop computers	Desktop and laptop computers, volume-based removable media (e.g., USB flash drives)	All types of end user devices	All types of end user devices
Data protected by encryption	All data on the media (data files, system files, residual data, and metadata)	All data in the volume (data files, system files, residual data, and metadata)	All data in the container (data files, residual data and metadata, but not system files)	Individual files/folders (data files only)
Mitigates threats involving loss or theft of devices?	Yes	Yes	Yes	Yes
Mitigates OS and application layer threats (such as malware and insider threats)?	No	If the data volume is being protected, it sometimes mitigates such threats.* If the data volume is not being protected, then there is no mitigation of these threats.	It sometimes mitigates such threats*	It sometimes mitigates such threats*
Potential impact to devices in case of solution failure	Loss of all data and device functionality	Loss of all data in volume; can cause loss of device functionality, depending on which volume is being protected	Loss of all data in container	Loss of all protected files/folders
Portability of encrypted information	Not portable	Not portable	Portable	Often portable

** These storage encryption technologies can only protect the files against some OS and application layer threats if the user has not been authenticated in this session to access the files. If a single sign-on solution is used, then generally the user is authenticated to the storage encryption technology during OS login, so the files are not protected against these threats once OS login occurs. If a separate authentication solution is used, the files are protected until that separate authentication is performed.*

When selecting storage encryption technologies, an organization should take into consideration the extent to which each technology will require the infrastructure and end user devices to be changed. For example, using some technologies requires deploying additional servers and installing software on the devices to be protected, while other technologies can use existing servers, as well as software built into the devices to be protected, such as FIPS approved encryption features built into the devices' operating systems. Generally, the more extensive the changes are to the infrastructure and devices, the more likely it is that the storage encryption technology will cause a loss of functionality or other problems with the devices. When evaluating solutions, organizations should compare the loss of functionality with the gain in security capabilities and decide if the tradeoff is acceptable. Technologies that require extensive changes to the infrastructure and end user devices should generally be used only when other technologies cannot meet the organization's needs.

The following are use cases that highlight the types of storage encryption technologies that may be suitable for certain situations. Each use case presents a brief scenario, including the threats that need to be mitigated, and then proposes possible solutions, both storage encryption technologies and alternate solutions (if applicable). Each use case only lists high-level solutions that may be feasible, and is not

intended to imply that other solutions are not possible or that these solutions are preferable to others. Each use case also omits security controls that are universal to the solutions, such as user awareness and general endpoint security (e.g., patching, antivirus software, access control, physical security of endpoint devices), as well as key management (for example, generating keys and securely deploying them to devices).

3.3.1 Use Case 1: Sharing a Laptop

Three users share a laptop. One of the users uses the laptop to access data that the other two users are not authorized to access. For this data, the major threats that the organization needs to mitigate are an insider threat from the other two users, and unauthorized disclosure of data from the loss or theft of the laptop. Possible solutions include the following:

- Implement volume, virtual disk, or file/folder encryption on the laptop. Protect the first user's data using the storage encryption software, with the authentication and cryptographic keys implemented so that only the first user, and not the other two users, can access the protected data. If there is concern about the first user always remembering where to save files, configure the laptop's access control so that the first user's data is all saved to a particular location, and protect that location with the storage encryption software.

- Store the data on external media, such as a flash drive or external hard drive, and use volume, virtual disk, or file/folder encryption to protect the media. The user needs to protect physical access to the media and to remember to save new or modified data to the media.

- Store the data on a remote system and give the first user access to the data through secured means (e.g., VPN). Provide the data in such a way that it is not saved to the laptop (e.g., the user views and modifies the remote data through a Web interface).

3.3.2 Use Case 2: Transferring Files Between Computers

A user edits documents using both a desktop PC at the organization's office and a personally owned computer at home. The user transfers documents between the computers on a daily basis using a USB flash drive. The two computers run different types of OSs. For the documents, the major threat that the organization needs to mitigate is unauthorized disclosure of data from loss or theft of the user's flash drive. Possible solutions include the following:

- Acquire and use a flash drive with self-contained storage encryption capabilities, such as encryption software and secure key storage.

- Acquire a volume, virtual disk, or file/folder encryption solution that will work on both PCs, and deploy it. Encrypt the documents using the solution and store the encrypted data on a flash drive.

3.3.3 Use Case 3: Sharing Data with Contractor

A user wants to provide a contractor with copies of large data sets on a daily basis because the contractor has no direct access to the system containing the data. The user will copy the data onto removable media for the contractor.[27] For this data, the major threat that the organization needs to mitigate is unauthorized disclosure of data from loss or theft of the removable media. Possible solutions include the following:

[27] Another use case, with similar possible solutions, is a user that needs to protect backups of a PC from loss or theft.

- Deploy virtual disk or file/folder encryption software to the user and contractor's computers. Encrypt the data using the software and burn the encrypted data onto CDs or DVDs.
- Acquire USB flash drives or external hard drives that have built-in storage encryption capabilities. Store the copies of the data on the encrypted drives.
- Acquire USB flash drives or external hard drives. Deploy virtual disk, volume, or file/folder encryption software to the user and contractor's computers. Encrypt the data using the software and store it on the drives.

3.3.4 Use Case 4: Traveling with a Laptop

A user occasionally travels on behalf of the organization and carries a laptop that contains sensitive data. For this data, the major threat that the organization needs to mitigate is unauthorized disclosure of data from the loss or theft of the laptop. Possible solutions include the following:

- Use the laptop's OS access control features to strictly limit where the user can save files. Implement volume, virtual disk, or file/folder encryption on the laptop to protect the user's files.
- Implement FDE on the laptop, and require pre-boot authentication.
- Provide the user with a loaner laptop when needed for travel. Protect the user's sensitive data on the laptop using either of the methods described above. When the user returns from travel, wipe and rebuild the loaner laptop to remove any traces of sensitive data from it. Using a loaner laptop in this way is particularly helpful if the laptop is being used in hostile environments, where the laptop is at greater risk of being compromised.

3.3.5 Use Case 5: Traveling with a Dual-Boot Laptop

A user frequently travels on behalf of the organization and carries a laptop that contains sensitive data. The laptop is dual-boot, using two OSs. For the user's data, the major threat that the organization needs to mitigate is unauthorized disclosure of data from the loss or theft of the laptop. Possible solutions include the following:

- Use the OS access control features of each OS to strictly limit where the user can save files. Implement volume, virtual disk, or file/folder encryption on both of the laptop's OSs to protect the user's files.
- Implement an FDE solution that supports dual-boot configurations.
- Convert the laptop to be single-boot, and access the second (removed) OS through a virtual machine run by the primary OS. See use case 4 for further information on protecting the laptop.

3.4 Storage Encryption Technology Management

Most storage encryption deployments are managed centrally. Centralized management is most often performed through special management utilities provided by the storage encryption vendor. If the storage encryption solution is built into the devices' OSs, then it could be managed through the mechanisms already in place to manage OS configurations. The capabilities of centralized management utilities for storage encryption technologies vary considerably. Examples of commonly implemented capabilities are as follows:

- Deploying storage encryption software to additional devices
- Updating storage encryption software (e.g., patching, upgrading)
- Configuring storage encryption software, such as specifying encryption algorithms and setting authentication policies (in some cases, the policies are specific for types of devices, groups of users, and/or individual users)
- Managing storage encryption authenticators and cryptographic keys[28]
- Collecting and reviewing storage encryption-related logs
- Recovering stored information from device failures
- Performing routine system maintenance
- Enabling the encryption of data and managing encrypted storage
  - For FDE, this involves encrypting the computer's hard drive. Some products allow the initial encryption to be done while the computer is in use, but it can cause a substantial performance impact if not configured properly, and additional hard drive space may be needed. Other products require that the computer not be in use while the drive is initially encrypted.
  - For volume encryption, this could involve either encrypting an existing volume, or creating a new volume, encrypting it, and then having the user add files to the volume as needed.
  - For virtual disk encryption, this simply involves creating a container. Files can then be added to the container by the user as needed.
  - For file/folder encryption, this could involve encrypting existing files, setting up an encrypted folder for future files, or establishing policies to automatically encrypt certain types of files.

Some storage encryption products, particularly ones intended for standalone deployment, can be managed locally. Local management is typically performed by a system administrator (for managed devices) or a user (for unmanaged devices) who has physical access to the device running the storage encryption technology. A common local management task is recovering data; many products allow users to recover their own data by running a recovery utility. For example, a device using FDE might experience a failure that prevents it from booting the OS; a user could run a recovery utility from the pre-boot environment or a CD to extract the data from the device.

Organizations may choose to deploy storage encryption without a centralized management capability and perform all management locally. This is generally acceptable for standalone deployments and very small-scale deployments, particularly ones that need to be done quickly, without waiting for a centralized management infrastructure to be implemented. However, for all other deployments, centralized management is recommended because it is more effective and efficient for most management tasks, including policy verification and enforcement, key management, authenticator management, and data recovery.

[28] Section 4 contains additional information on cryptography, key management, and authentication.

4. Storage Encryption Technology Planning and Implementation

This section discusses considerations for planning and implementing storage encryption technologies for end user devices. As with any new technology deployment, storage encryption technology planning and implementation should be addressed in a phased approach. A successful deployment can be achieved by following a clear, step-by-step planning and implementation process. The use of a phased approach for deployment can minimize unforeseen issues and identify potential pitfalls early in the process. This model also allows for incorporating advances in new technology and adapting the technology to the ever-changing enterprise. The following is an example of planning and implementation phases:

1. **Identify Needs.** The first phase involves identifying the needs to encrypt storage on end user devices, determining which devices and data need protection, and identifying related requirements (e.g., minimum performance). This phase also involves determining how that need can best be met (e.g., FDE, virtual disk encryption) and deciding where and how the security should be implemented.

2. **Design the Solution.** The second phase involves all facets of designing the solution. Examples include architectural considerations, authentication methods, cryptography policy, and supporting security controls.

3. **Implement and Test a Prototype.** The next phase involves implementing and testing a prototype of the designed solution in a lab or test environment. The primary goals of the testing are to evaluate the functionality, performance, scalability, and security of the solution, and to identify any issues with the components, such as interoperability issues.

4. **Deploy the Solution.** Once the testing is completed and all issues are resolved, the next phase includes the gradual deployment of the storage encryption technology throughout the enterprise.

5. **Manage the Solution.** After the solution has been deployed, it is managed throughout its lifecycle. Management includes maintenance of the storage encryption components and support for operational issues. The lifecycle process is repeated when enhancements or significant changes need to be incorporated into the solution.

This document does not describe the planning and implementation process in depth because the same basic steps are performed for any security technology. For example, the document assumes that the organization has already determined what information needs to be protected and assigned an impact level from FIPS 199 to the information. This section only highlights those considerations that are particular to storage encryption for end user devices.

4.1 Identify Needs

The purpose of this phase is to identify the needs to protect information stored on end user devices and determine how those needs can best be met. Requirements specific to storage encryption that should be considered include the following:

- **External Requirements.** The organization may be subject to oversight or review by another organization that requires storage encryption. An example is a legal requirement to protect stored PII.

- **System and Network Environments.** It is important to understand the characteristics of the organization's system and network environments so that storage encryption solutions can be selected that will be compatible with them and able to provide the necessary protection. Aspects to consider include the following:

- The characteristics of the devices that need protection, especially the OSs, applications, and filesystems they use, and their hardware capabilities and characteristics

- The technical attributes of the interfaces of other systems with which the storage encryption solution might be integrated, such as authentication services, centralized logging servers and security information and event management (SIEM) software, and patch management software

- **Support Limitations.** The organization should identify any negative impacts that storage encryption technologies could have on existing vendor support mechanisms. For example, installing a storage encryption technology onto an end user device could violate the terms of a support contract for existing software on the end user device or void a warranty for another product used on or with the end user device.

The outcome of the organization's analysis should be a determination of which files or types of files need to be encrypted and which types of threats the storage encryption software should protect against, stating the concerns as specifically as possible. For example, the organization may decide to encrypt sensitive information on all devices used outside the organization's facilities and to encrypt certain types of sensitive information on devices used from any location.

The analysis should also lead to the identification of the type or types of storage encryption technologies that can meet the organization's security needs. Another outcome of the analysis is the documentation of the requirements for the storage encryption technologies themselves, including security capabilities (e.g., authentication, cryptography, key management), performance requirements, management requirements (including reliability, interoperability, scalability), the security of the technology itself, usability (by both administrators and users), and maintenance requirements (such as applying updates).

In most cases, a single storage encryption product cannot meet all of the organization's identified needs. For example, the organization may need to protect information on devices running several different OSs, yet no appropriate product can work on all those platforms. Some devices might also not meet the minimum hardware requirements for storage encryption products. Organizations can solve this problem in several ways, such as acquiring multiple products, using multiple types of storage encryption technologies, replacing older devices, or identifying compensating controls to be used instead of storage encryption that provide the same level of protection. Appendix A discusses some potential alternatives to storage encryption. Organizations should ensure that effective solutions are identified for all the types of end user devices that need their stored information protected, if possible, and that a waiver process is created for unusual cases that cannot be addressed by the identified solutions.

4.2 Design the Solution

Once the needs have been identified and the appropriate type(s) of storage encryption technology have been chosen, the next phase is to design a solution that meets the needs. If these design decisions are incorrect, then the storage encryption implementation will be more susceptible to compromise. Major aspects of solution design that are particularly important for storage encryption are as follows:

- **Cryptography.** Encryption and integrity protection algorithms must be selected, as well as the key strength for algorithms that support multiple key lengths. Key management and protection is another important component of solution design. Section 4.2.1 contains additional cryptography information.

- **Authentication.** Authentication methods must be chosen for users and administrators. Decisions also need to be made regarding the protection of the authenticators themselves. Section 4.2.2 contains a more detailed discussion of authentication.

- **Solution architecture.** The architecture of the storage encryption implementation refers to the selection of devices and software to provide storage encryption services and the placement of centralized elements within the existing network infrastructure, such as authentication credential servers, Web servers for self-service recovery, and management servers. Each end user device must have hardware and/or software that provides protection for the stored information. Designing the architecture includes component placement, redundancy, reliability, and interoperability.

- **Other security controls.** These support and complement the storage encryption implementation. For example, organizations should have policies regarding acceptable usage of storage encryption technologies. Organizations may also set minimum security standards for end user devices, such as mandatory host hardening measures and patch levels, and specify security controls that must be employed, such as host-based personal firewalls, antivirus software, and antispyware software.

- **Minimum requirements for end user devices.** The minimum requirements for the hardware, OS, and supporting software should be defined. They should be based on the requirements supplied by the product vendor and the organization's performance requirements.

Another aspect of solution design is planning the logistics of the solution's deployment. For example, the organization may need to replace devices that do not meet minimum requirements or run on a platform that the organization will not support for storage encryption. This could cause out-of-cycle upgrades or replacements of hardware, OSs, and supporting software. Another logistical consideration is how the solution will be deployed to end user devices. If devices need to be updated locally, such as upgrading the OS, replacing the hard drive, backing up user files, or installing storage encryption software, then organizations need to plan who will perform these actions and when and where the work will be done. Some organizations may need to set up staging areas and get additional personnel to perform this work.

4.2.1 Cryptography

Storage encryption technologies use one or more cryptographic keys to encrypt and decrypt the data that they protect. The number of keys and the types of keys used are product and implementation-dependent. For example, public key cryptography uses a pair of keys, and symmetric cryptography uses a single key. Some products support the use of a recovery key that can be used to recover the encrypted data if the regular key is lost. Also, some technologies permit encrypted storage to be shared by multiple users, which could be enabled by having a different key for each user. Often, users' keys are not directly used to decrypt their stored data; instead, those keys are used to decrypt another key, which in turn is used to decrypt the stored data.

If a key is lost or damaged, it may not be possible to recover the encrypted data. Therefore, organizations need to ensure that all keys used in a storage encryption solution are secured and managed properly to support the security of the solution. Organizations should perform extensive planning of key management processes, procedures, and technologies before implementing storage encryption technologies. This planning should include all aspects of key management, including key generation, use, storage, recovery, and destruction.[29] Organizations should carefully consider how key management practices can support the recovery of encrypted data if a key is inadvertently destroyed or otherwise becomes unavailable (such as a user unexpectedly leaving an organization or losing a cryptographic token containing a key). An example of recovery preparation is storing duplicates of keys in a centralized, secured key repository or on physically secured removable media. Organizations planning on encrypting

[29] NIST SP 800-57, *Recommendation for Key Management*, provides detailed information on key management planning, algorithm selection and appropriate key sizes, cryptographic policy, and cryptographic module selection. Organizations may be able to use the same or similar key management processes for end user devices' storage encryption, virtual private network (VPN) clients, and wireless client configuration.

removable media also need to consider how changing keys will affect access to encrypted storage on the media and develop feasible solutions, such as retaining the previous keys in case they are needed.

Another decision that may need to be made is where the local keys should be stored. For some encryption technologies, such as FDE and many file/folder encryption products, there are often several options for key location, including the local hard drive, a USB flash drive, a cryptographic token, or a Trusted Platform Module (TPM) chip.[30] Some products also permit keys to be stored on a centralized server and retrieved automatically after the user authenticates successfully. For volume and virtual disk encryption, the main encryption key is often stored encrypted within the volume or container itself. Some storage encryption products do not store a key; instead, they perform a cryptographic hash function on the password entered by the user and use that hash as the key.

Organizations need to ensure that access to keys (other than those intended to be available to others, such as public keys) is properly restricted. Storage encryption solutions should require the use of one or more authentication mechanisms, such as passwords, smart cards, and cryptographic tokens, to decrypt or otherwise gain access to a storage encryption key. The keys themselves should be logically secured (e.g., encrypted) or physically secured (e.g., stored in a tamper-resistant cryptographic token). The authenticators used to retrieve keys should also be secured properly.

In addition to key management, there are several other aspects of cryptography that need to be considered when planning a storage encryption solution. Setting the cryptography policy involves choosing encryption and integrity protection algorithms and key lengths.[31] Federal agencies must use FIPS-approved algorithms contained in validated cryptographic modules.[32] Whenever possible, AES[33] should be used for the encryption algorithm because of its strength and speed. Several FIPS-approved algorithms are available for integrity checking, including HMAC-SHA, Cipher-Based Message Authentication Code (CMAC), and Counter with Cipher Block Chaining-Message Authentication Code (CCM).[34] Organizations should consider how easily the solution can be updated when stronger algorithms and key sizes become available in the future.

4.2.2 Authentication

There are two types of authentication important to storage encryption. Administrators authenticate so that they can perform storage encryption management functions, including reconfiguring and updating encryption software, managing user accounts, and recovering encrypted data.[35] Users authenticate so that they can access encrypted information. If a single authenticator is used (often a user ID and password, sometimes a token), that authenticator typically grants the storage encryption software access to the key used to encrypt and decrypt the stored information. If two-factor authentication is used, typically one of

---

30 A *TPM chip* is a tamper-resistant integrated circuit built into some motherboards that can perform cryptographic operations (including key generation) and protect small amounts of sensitive information, such as passwords and cryptographic keys. As of this writing, no FIPS-approved TPM chips are yet available.

31 NIST SP 800-21, Second Edition, *Guideline for Implementing Cryptography in the Federal Government*, presents guidelines for selecting, specifying, employing, and evaluating cryptographic protection mechanisms in Federal information systems. It defines a process for selecting cryptographic products and discusses implementation issues, including solution management, key management, and authentication. NIST SP 800-21 is available at http://csrc.nist.gov/publications/nistpubs/.

32 The Cryptographic Module Validation Program (CMVP) at NIST coordinates FIPS 140-2 testing; the CMVP Web site is located at http://csrc.nist.gov/cryptval/. See http://csrc.nist.gov/cryptval/des.htm for information on FIPS-approved symmetric key algorithms, and http://csrc.nist.gov/cryptval/dss.htm for information on digital signature algorithms. FIPS 140-2, *Security Requirements for Cryptographic Modules*, is available at http://csrc.nist.gov/publications/fips/.

33 For more information, read FIPS 197, *Advanced Encryption Standard (AES)*, at http://csrc.nist.gov/publications/fips/.

34 Additional information on these algorithms is available at http://csrc.nist.gov/CryptoToolkit/modes/.

35 The term "administrators" is used generically to refer to the individuals responsible for managing the storage encryption solution. Some organizations use more specific terms for these individuals, such as "cryptographic officers".

the factors grants access to information secured in another factor, which is then used to gain access to the storage encryption key. For example, a PIN or password could be used to retrieve a key from a smart card or cryptographic token; that key could then be used to decrypt the storage encryption key.[36]

Some storage encryption products allow the use of multiple user IDs on a single device. If the IDs are tied to a single storage encryption key, then each user can access the same protected information. If each ID is linked to a separate key, then the access is dependent on how the keys are used—for example, a container could be encrypted using a single key so that only one user can access it, or encrypted using several keys so that users can share the contents of the container.

For storage encryption authentication, organizations often want to leverage existing enterprise authentication solutions (e.g., Active Directory, RADIUS, PKI, Personal Identity Verification [PIV] cards) instead of adding another authenticator for users. Generally, using an existing authentication solution is acceptable only if it provides multi-factor authentication. Using a single-factor authenticator for multiple purposes significantly weakens the protection that authentication provides.[37] For example, reusing a user's OS password for pre-boot authentication in an FDE deployment would allow an attacker to learn only a single password to gain full access to the device's information. The password could potentially be acquired through technical methods, such as infecting the device with malware, or through physical means, such as watching a user type in a password in a public location. Another example is having a volume, virtual disk, or file/folder encryption product use the OS's authentication for single sign-on (SSO). Once a user authenticates to the OS at login, the user can access the encrypted files without further authentication, so the security of the solution is heavily dependent on the strength of the OS authenticator. Organizations should carefully consider the security implications of using the same single-factor authenticator for multiple purposes. In particular, organizations should not use email passwords and other passwords sometimes transmitted in plaintext as single-factor authenticators for storage encryption. Also, organizations concerned about targeted attacks, such as someone stealing a particular laptop to gain access to a specific user's data, should not use only passwords for storage encryption authentication because of the relative ease of capturing a user's password (e.g., watching a password being typed).

Organizations also need to ensure that the storage encryption authenticators are protected properly. This includes both technical mechanisms, such as encrypting passwords or storing cryptographic hashes of passwords, and operational and management mechanisms. For example, policy should state and users should be made aware that authenticators should not be stored in proximity of the end user device (e.g., a password should not be on a piece of paper in a laptop case), and that for two-factor authentication, multiple authenticators should not be stored with each other (e.g., a password or PIN should not be written on the back of a hardware token).

Because authentication controls access to storage encryption keys, the loss of authenticators can prevent access to the encrypted data. Organizations should determine how the loss of authenticators (both user and administrator-level) will be handled before implementing storage encryption. Most products offer recovery mechanisms for password-based user authentication. For example, a user that has forgotten a password chooses a recovery option on the protected computer. The computer provides a special code to the user, who then uses another computer to access the organization's storage encryption recovery Web

---

[36] The following authentication methods for storage encryption are listed from weakest to strongest based on their general effectiveness: single sign-on, unique password or PIN, token, token with unique password or PIN. See NIST SP 800-63, *Electronic Authentication Guideline*, for additional information (http://csrc.nist.gov/publications/nistpubs/).

[37] In many cases, single-factor authentication will not be an option because the sensitivity of the data being protected will necessitate the use of multi-factor authentication. Using the same single-factor authenticator for storage encryption and other purposes may be acceptable if appropriate compensating controls are used, such as the device using storage encryption residing only in physically secured office space.

site. The user provides the code to the Web site and gives proof of identity, such as answering questions about personal preferences (e.g., favorite color) that the user has previously configured. The Web site then provides the user's password or a one-time recovery code that the user enters into the computer to regain access. A similar process can also be performed by having users call a help desk instead of accessing a particular Web site.

For user authentication methods other than password-based, recovery is often more difficult, especially if the user is not at the organization's facilities. Some storage encryption products allow the password-based authentication recovery mechanisms to be used and permit the user to temporarily use password-based authentication. However, because this is generally a reduction in the strength of authentication, many organizations do not permit its use. This means that a loss of authenticator could cause an extended loss of availability to the data, until the user can receive a new authenticator (e.g., smart card, cryptographic token) and an administrator can configure the device to use the new authenticator.

Recovery mechanisms increase the availability of the storage encryption solution for individual users, but they can also increase the likelihood that an attacker can gain unauthorized access to encrypted storage by abusing the recovery mechanisms. Organizations should consider the tradeoff between availability and security when selecting and planning recovery mechanisms.

Some storage encryption products also offer protection against authentication-guessing attempts. For example, if there are too many consecutive failed authentication attempts, some products can either lock the computer for a period of time or increase the delay between attempts. In particularly high-security situations, some products can be configured so that too many failed attempts causes the product to wipe all the protected data from the device. This approach strongly favors security over functionality.

4.3 Implement and Test Prototype

After the solution has been designed, the next step is to implement and test a prototype of the design. Ideally, implementation and testing should first be performed on lab or test devices. Only implementations in final testing should be conducted on production devices. Aspects of the solution to evaluate include the following:

- **Protection.** Each type of information that needs protection should be protected in accordance with the information gathered during the Identify Needs phase. This should be verified by using forensic tools to confirm that the information is encrypted. For devices that use FDE and offer hibernation, standby, or other "suspend" modes, encryption should be verified in each mode; if the mode does not write the contents of memory out to disk and encrypt it, then the information may be readily available unencrypted.[38]

- **Authentication.** Performing robust testing of authentication is important, especially for more complex authentication solutions that depend on centralized authentication services; a loss of those services could cause a loss of storage encryption services as well.

- **OS and Application Compatibility.** The solution should not break or interfere with the use of existing OS configurations and software applications. Examples of applications that may be particularly problematic are, for FDE, disk-level software tools, asset management software, and dual-boot configurations, and for all storage encryption technologies, backup utilities, forensic tools, and other storage encryption programs.

[38] This can be addressed by configuring the device not to use modes that maintain the data in an unencrypted format.

- **Management.** Administrators should be able to configure and manage all components of the solution effectively and securely. It is particularly important to evaluate the ease of deployment and configuration, including how easily the solution can be managed as the solution is scaled to larger deployments. Another concern is the ability of administrators to disable configuration options so that users cannot circumvent the intended security. Management concerns should include the effects of patching/upgrading software, changing software settings (e.g., changing cryptographic algorithms or key sizes), uninstalling or disabling encryption software, changing encryption/decryption keys, and changing user or administrator passwords.

- **Logging.** The logging and data management functions should function properly in accordance with the organization's policies and strategies.

- **Performance.** The solution should be able to provide adequate performance during normal and peak usage. Testing should incorporate a variety of devices, OSs, and applications, especially those that are most likely to be affected by performance issues, such as those that manipulate large files.

- **Security of the Implementation.** The storage encryption implementation itself may contain vulnerabilities and weaknesses that attackers could exploit. Organizations with high security needs may want to perform extensive vulnerability assessments against the storage encryption components. Another common security concern is the security of the authenticators and cryptographic keys.

- **Recovery.** The solution should be tested to determine how well it can recover from failures, such as lost or forgotten authenticators, lost keys, device hardware or software failure/damage, and power loss.

- **Interoperability.** For a solution that will protect removable media that will be used on multiple devices, the organization should ensure that information encrypted on the media by one device can be decrypted by another device after authenticating successfully.

- **Operational Impacts.** Organizations should determine how the solution might impact operations, such as impeding technical support and incident response actions involving end user devices.

Organizations should consider implementing the components in a test environment first, instead of a production environment, to reduce the likelihood of implementation problems disrupting the production environment. When the components are being deployed into production, organizations should initially use encryption on a small number of hosts. Deploying it to many hosts at once might overwhelm the management servers or identify other bottlenecks through loss of availability. Many of the problems that occur are likely to occur on multiple hosts, so it is helpful to identify such problems either during the testing process or when deploying the first hosts, so that those problems can be addressed before widespread deployment. A phased deployment is also helpful in identifying potential problems with scalability.

Actions that may be prudent to perform before installing storage encryption software on end user devices include the following:

- Ensure that any files to be encrypted can be restored. Examples include backing up user files and having a disk image for the computer's OS.

- Replace hardware components (e.g., replace an old hard drive) or the whole device if necessary (e.g., equipment that is considered too slow or unreliable).

- Ensure that the OS is secured properly, including that it is fully patched and that other necessary security controls, such as antivirus software, are installed and configured properly. If the OS is not secured properly, the device is more likely to be compromised, which could weaken the protection provided by the storage encryption solution.

- Scan the device for malware and either remove any malware that is detected or rebuild the device.

4.4 Deploy the Solution

Once testing is complete and any issues have been resolved, the next phase of the planning and implementation model involves deploying the solution. A prudent strategy is to gradually migrate devices and users to the new solution. The phased deployment provides administrators an opportunity to evaluate the impact of the solution and resolve issues prior to enterprise-wide deployment. It also provides time for the IT staff (e.g., system administrators, help desk) and users to be trained.

Most of the issues that can occur during deployment are the same types of issues that occur during any large IT deployment. In addition to potential problems described earlier in this publication, another typical issue is that storage encryption technologies might not work properly on some devices because of incompatibilities with particular hardware configurations.

4.5 Manage the Solution

The last phase of the planning and implementation model is the longest lasting. Managing the solution involves operating the deployed solution and maintaining the security storage architecture, policies, software, and other solution components. Examples of typical actions are as follows:

- Testing and applying patches to storage encryption software. It is beneficial to have at least one host (one for each type of platform) that is used strictly for testing updates. This can help to identify possible conflicts between an update and the normal functions of devices. Software updates should be tested and deployed using the same practices that would be used for updating any other major security controls, such as antivirus software.

- Deploying storage encryption technologies to additional types of devices

- Configuring additional devices to use the technologies

- Performing key management duties (e.g., issuing new credentials, revoking credentials for compromised systems or departing users)

- Performing recovery actions (e.g., regaining access to encrypted data when the authenticator has been lost or the storage media has been damaged)

- Adapting the policies as requirements change. An example is switching to a stronger encryption algorithm or increasing the key size.

- Monitoring the storage encryption components for operational and security issues

- Periodically performing testing to verify that storage encryption is functioning properly

- Performing regular vulnerability assessments

- Receiving notifications from vendors of security problems with storage encryption components, and responding appropriately to those notifications

- Preparing devices for retirement or disposal. Devices and media that use storage encryption technologies should be sanitized or destroyed, even for devices using FDE. As mentioned in Section 3.2, relying on storage encryption to protect data without regularly maintaining the encryption solution is not recommended. Another problem with relying on storage encryption to protect data on retired or disposed devices is that all copies of all keys used for storage encryption would need to be destroyed, which may be very difficult.

User files on the device should be backed up before major maintenance actions are performed, such as installing or upgrading storage encryption software and changing encryption algorithms or key sizes.

Appendix A—Alternatives to Encrypting Storage on End User Devices

Section 3 describes commonly used technologies for storage encryption—full disk, virtual disk and volume, and file/folder encryption. Organizations should not feel compelled to use only these methods to encrypt stored information; there are many other acceptable methods. The following are some examples:

- Applications can encrypt the information that they store. For example, a commercial off-the-shelf (COTS) backup utility might be capable of encrypting its backups, and a compression utility might have an option to encrypt archives that a user creates. Another example is a database that can be configured to encrypt fields that contain sensitive information. An application could also store sensitive information in an alternate format, such as cryptographic hashes of passwords instead of the passwords themselves.

- Sensitive information could be accessed only through a virtual machine and stored as part of the virtual machine. If the virtual machine software itself does not provide an encryption capability, the virtual machine data, which is a single file, could be protected through storage encryption software.

In some cases, organizations may decide that the best way to address the problem of protecting sensitive information on end user devices is not to store the information on the devices. Examples of how this might be implemented include the following:

- Preventing access to sensitive information from higher-risk devices, such as mobile devices

- Using a thin client solution, such as terminal services, a thin Web-based application, or a portal, to access the information, and configuring the thin client solution to prohibit file transfers of the sensitive information to the end user device

- Configuring the organization's devices (including desktop computers) to prevent writing sensitive information to removable media, such as CDs or USB flash drives, unless the information is properly encrypted

- Permitting users to access files or databases containing sensitive information only through well-secured applications that restrict access as tightly as possible. For example, suppose that an organization has a database containing thousands of records on employees' benefits. Instead of allowing a user to have full and direct access to the database, which could allow the user to transfer all the database records to the user's device, the organization could permit the user to access only the necessary records and record fields. If the user only needs access to general demographic information, and does not need to access any information related to the employees' identities, then the user would not be able to access any sensitive information.

- Removing unneeded sensitive information from files or databases.

Organizations should be aware that if sensitive information can be viewed from end user devices, the information is still at some risk of exposure, even if it is not stored there. The information could be recorded in screen captures, printed, or monitored by malware, for example.

Organizations should also be aware that the use of general access control mechanisms is typically insufficient to protect sensitive information on end user devices. For example, a password generally cannot be used instead of cryptography to protect stored information. Although requiring a BIOS password can prevent an attacker from booting a computer regularly, the attacker could still access the information by placing the storage media in a different computer. Using an OS password is also

ineffective because forensic tools can examine the storage media directly. OS controls such as access control lists may deter users from accessing each other's files, but they can also be circumvented by using forensic tools. Properly designed and implemented encrypted storage cannot be decrypted by forensic tools.

Appendix B—Glossary

Selected terms used in the publication are defined below.

Container: The file used by a virtual disk encryption technology to encompass and protect other files.

End User Device: A personal computer (desktop or laptop), consumer device (e.g., personal digital assistant [PDA], smart phone), or removable storage media (e.g., USB flash drive, memory card, external hard drive, writeable CD or DVD) that can store information.

File: A collection of information logically grouped into a single entity and referenced by a unique name, such as a filename.

File Encryption: The process of encrypting individual files on a storage medium and permitting access to the encrypted data only after proper authentication is provided.

Filesystem: A mechanism for naming, storing, organizing, and accessing files on logical volumes.

Folder: An organizational structure used by a filesystem to group files.

Folder Encryption: The process of encrypting individual folders on a storage medium and permitting access to the encrypted files within the folders only after proper authentication is provided.

Full Disk Encryption (FDE): The process of encrypting all the data on the hard drive used to boot a computer, including the computer's operating system, and permitting access to the data only after successful authentication with the full disk encryption product.

Malware: A program that is inserted into a system, usually covertly, with the intent of compromising the confidentiality, integrity, or availability of the victim's data, applications, or operating system.

Master Boot Record (MBR): A special region on bootable media that determines which software (e.g., operating system, utility) will be run when the computer boots from the media.

Metadata: Data about data; in the context of a filesystem, information regarding files and folders themselves, such as file and folder names, creation dates and times, and sizes.

Partitioning: The act of logically dividing a media into portions that function as separate units.

Pre-Boot Authentication (PBA): The process of requiring a user to authenticate successfully before decrypting and booting an operating system.

Residual Data: Data from deleted files or earlier versions of existing files.

Sector: The smallest unit that can be accessed on media.

Storage Security: The process of allowing only authorized parties to access stored information.

Trusted Platform Module (TPM) Chip: A tamper-resistant integrated circuit built into some computer motherboards that can perform cryptographic operations (including key generation) and protect small amounts of sensitive information, such as passwords and cryptographic keys.

Virtual Disk Encryption: The process of encrypting a container, which can hold many files and folders, and permitting access to the data within the container only after proper authentication is provided.

Volume: A logical unit of storage comprising a filesystem.

Volume Encryption: The process of encrypting an entire volume and permitting access to the data on the volume only after proper authentication is provided.

Whole Disk Encryption: See "Full Disk Encryption".

Appendix C—Acronyms

Selected acronyms used in the publication are defined below.

AES	Advanced Encryption Standard
BIOS	Basic Input/Output System
CAVP	Cryptographic Algorithm Validation Program
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CMAC	Cipher-Based Message Authentication Code
CMVP	Cryptographic Module Validation Program
COTS	Commercial Off-the-Shelf
DRM	Digital Rights Management
EFS	Encrypting File System
EPHI	Electronic Protected Health Information
FDE	Full Disk Encryption
FIPS	Federal Information Processing Standards
FISMA	Federal Information Security Management Act
GLBA	Gramm-Leach-Bliley Act
HIPAA	Health Insurance Portability and Accountability Act
HMAC	Keyed-Hash Message Authentication Code
IT	Information Technology
ITL	Information Technology Laboratory
KB	Kilobyte
MBR	Master Boot Record
NIST	National Institute of Standards and Technology
NTFS	New Technology File System
NVD	National Vulnerability Database
OMB	Office of Management and Budget
OS	Operating System
PBA	Pre-Boot Authentication
PBE	Pre-Boot Environment
PDA	Personal Digital Assistant
PII	Personally Identifiable Information
PIN	Personal Identification Number
PKI	Public Key Infrastructure
RADIUS	Remote Authentication Dial In User Service

SHA	Secure Hash Algorithm
SIEM	Security Information and Event Management
SP	Special Publication
SSO	Single Sign-On
TPM	Trusted Platform Module
USB	Universal Serial Bus
VPN	Virtual Private Network

Appendix D—Tools and Resources

The lists below provide examples of tools and resources that may be helpful.

Resource Web Sites

Organization	URL
Computer Forensics Tool Testing (CFTT) Project	http://www.cftt.nist.gov/
Cryptographic Algorithm Validation Program (CAVP)	http://csrc.nist.gov/groups/STM/cavp/index.html
Cryptographic Module Validation Program (CMVP)	http://csrc.nist.gov/groups/STM/cmvp/index.html
Full Disk Encryption mailing list and discussion group	http://www.full-disc-encryption.com/
Modes of Operation for Symmetric Key Block Ciphers	http://csrc.nist.gov/CryptoToolkit/modes/

Resource Documents

Document Title	URL
FIPS 140-2, *Security Requirements for Cryptographic Modules*	http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf
FIPS 180-2, *Secure Hash Standard*	http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf
FIPS 197, *Advanced Encryption Standard (AES)*	http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
NIST SP 800-21-1, *Guideline for Implementing Cryptography in the Federal Government*	http://csrc.nist.gov/publications/nistpubs/800-21-1/sp800-21-1_Dec2005.pdf
NIST SP 800-32, *Introduction to Public Key Technology and the Federal PKI Infrastructure*	http://csrc.nist.gov/publications/nistpubs/800-32/sp800-32.pdf
NIST SP 800-38A, *Recommendation for Block Cipher Modes of Operation-Methods and Techniques*	http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf
NIST SP 800-38B, *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*	http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf
NIST SP 800-38C, *Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*	http://csrc.nist.gov/publications/nistpubs/800-38C/SP800-38C.pdf
NIST SP 800-38D (DRAFT), *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) for Confidentiality and Authentication*	http://csrc.nist.gov/publications/drafts.html
NIST SP 800-53 Revision 1, *Recommended Security Controls for Federal Information Systems*	http://csrc.nist.gov/publications/nistpubs/800-53-Rev1/800-53-rev1-final-clean-sz.pdf
NIST SP 800-57, *Recommendation for Key Management—Part 1: General*	http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf
NIST SP 800-57, *Recommendation for Key Management—Part 2: Best Practices for Key Management Operation*	http://csrc.nist.gov/publications/nistpubs/800-57/SP800-57-Part2.pdf
NIST SP 800-63, *Electronic Authentication Guideline*	http://csrc.nist.gov/publications/nistpubs/800-63/SP800-63V1_0_2.pdf
NIST SP 800-67, *Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher*	http://csrc.nist.gov/publications/nistpubs/800-67/SP800-67.pdf

NIST Special Publication 800-52

Guidelines for the Selection and Use of Transport Layer Security (TLS) Implementations

NIST
National Institute of Standards and Technology
Technology Administration
U.S. Department of Commerce

C. Michael Chernick, Charles Edington III, Matthew J. Fanto, Rob Rosenthal

COMPUTER SECURITY

June 2005

NIST Special Publication 800-52

Guidelines for the Selection and Use of Transport Layer Security (TLS) Implementations

Recommendations of the National Institute of Standards and Technology

C. Michael Chernick, Charles Edington III, Matthew J. Fanto, Rob Rosenthal

C O M P U T E R S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

June 2005

U.S. Department of Commerce
Donald L. Evans, Secretary

Technology Administration
Phillip J. Bond, Under Secretary of Commerce for Technology

National Institute of Standards and Technology
Arden L. Bement, Jr., Director

Table of Contents

Tables

Figures

Executive Summary

Office of Management and Budget (OMB) Circular A-130, *Management of Federal Information Resources,* requires managers of publicly accessible information repositories or dissemination systems that contain sensitive but unclassified data to ensure sensitive data is protected commensurate with the risk and magnitude of the harm that would result from the loss, misuse, or unauthorized access to or modification of such data. Given the nature of interconnected networks and the use of the Internet to share information, protection of this sensitive data can become difficult if proper mechanisms are not employed to protect the data. Transport layer security (TLS) provides such a mechanism to protect sensitive data during electronic dissemination across the Internet.

TLS is a protocol created to provide authentication, confidentiality and data integrity between two communicating applications. TLS is based on a precursor protocol called "The Secure Sockets Layer Version 3.0" (SSL 3.0) and is considered to be an improvement to SSL 3.0. SSL is specified in an expired Internet Draft working document of the Internet Engineering Task Force. Transport Layer Security Version 1 (TLS 1.0) specification is an Internet Request for Comments [RFC2246]. Each document specifies a similar protocol that provides security services over the Internet. While TLS 1.0 is based on SSL 3.0, and the differences are not dramatic; they are significant enough that TLS 1.0 and SSL 3.0 do not interoperate. This Special Publication provides guidance to the selection and implementation of the TLS protocol while making effective use of Federal Information Processing Standards (FIPS) approved cryptographic algorithms, and suggests that TLS 1.0 configured with FIPS based cipher suites is the appropriate secure transport protocol.[1] Use of the recommendations provided in this Special Publication would promote:

- More consistent use of authentication, confidentiality and integrity mechanisms for the protection of information transport across the Internet;
- Consistent use of recommended cipher suites that encompass FIPS algorithms and open source technology; and
- Informed decisions by system administrators and managers in the integration of transport layer security implementations.

While these Guidelines are primarily designed for Federal users and system administrators to adequately protect sensitive but unclassified Federal Government data against serious threats on the Internet, they may also be used within closed network environments to segregate data. (The client-server model and security services discussed also apply in these situations). This Special Publication does not supercede any existing NIST publication and should be used in conjunction with existing policies and procedures.

---

[1] While SSL 3.0 is the most secure of the SSL protocol versions, it is not approved for use in the protection of Federal information because it relies in part on the use of cryptographic algorithms that are not FIPS-Approved. TLS when properly configured is approved for the protection of Federal information.

Reports on Information Security Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive, unclassified information in Federal computer systems. This Special Publication 800-series reports on ITL's research, guidance, and outreach efforts in computer security and its collaborative activities with industry, government, and academic organizations.

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

National Institute of Standards and Technology Special Publication 800-52
Natl. Inst. Stand. Technol. Spec. Publ. 800-52, 33 pages (June 2005)
CODEN: NSPUE2

Acknowledgments

The authors, Michael Chernick and Matt Fanto of NIST, and Charles Edington and Robert Rosenthal of Booz Allen Hamilton (BAH) would like to thank the many people who assisted with the development of this document. In particular we would like to acknowledge Marcus Sills of BAH who aided with researching the complexities of the TLS protocol and Elaine Barker of NIST who gave guidance on random number generation.

1 Introduction

Today, many World Wide Web browsers and server applications rely on secure SSL and TLS communications to protect sensitive data transmitted through the Internet. Many books such as [Rescorla01], [Comer00], and [Hall00] describe the Internet's client-server model and communication protocol design principles. None guide Federal users and system administrators to adequately protect sensitive but unclassified Federal Government data against the most serious threats on the World Wide Web – eavesdropping, data tampering and message forgery. Other books such as [Adams99] and [Housley01] as well as technical journal articles (e.g., [Polk03]) and NIST publications (e.g., [SP800-32]) describe how Public Key Infrastructure (PKI) can be used to protect information in the Internet.

It is assumed that the reader of these Guidelines is somewhat familiar with the ISO seven-layer model communications model (also known as the seven-layer stack) [7498], as well as the Internet and public key infrastructure concepts, including, for example, X.509 certificates. If not, the reader may refer to the references cited above in the first paragraph of this introduction for further explanations of background concepts that cannot be fully explained in these Guidelines.

These Guidelines briefly introduce computer communications architectural concepts. The Guidelines place the responsibility for communication security at the Transport layer of the OSI seven-layer communications stack, not within the application itself. Protection of sensitive but unclassified Government information can adequately be accomplished at this layer when appropriate protocol options are selected and used by clients and servers relying on transport layer security.

Unfortunately, security is not a single property possessed by a single protocol. Rather, security includes a complex set of related properties that together provide the required information assurance characteristics and information protection services. Security requirements are usually derived from a risk assessment to the threats or attacks an adversary is likely to mount against a system. The adversary is likely to take advantage of implementation vulnerabilities found in many system components including computer operating systems, application software systems, and the computer networks that interconnect them. These guidelines focus only on security within the network, and they focus directly on the small portion of the network communications stack that is referred to as the transport layer.

Usually, the best defense against telecommunications attacks is to deploy security services implemented with mechanisms specified in standards that are thoroughly vetted in the public domain and rigorously tested by third party laboratories, by vendors, and by users of commercial off-the-shelf products.

Three services that most often address network user security requirements are confidentiality, message integrity and authentication. A confidentiality service provides assurance that data is kept secret, preventing eavesdropping. A message integrity service provides confirmation that data modification is always detected thus preventing undetected deletion, addition, or modification of data. An authentication service provides assurance of the sender or receiver's identity, thereby preventing forgery.

2 Security in a Layered Communications Architecture

The layering of computer communication protocols into a protocol stack (as defined by the OSI Seven Layer Model [7498]) enables developers to design new communication systems using already defined protocols and specific communication requirements within each layer of the stack. Each layer of the transmitting system communicates with the corresponding layer on the receiving system(s). Within this communications stack, the functionality of each layer is, in theory, independent of each other layer. Placement of security services and implementation of the security mechanisms within the stack is specific to each individual layer of the stack. Since the OSI Seven Layer Model does not explicitly define where security services are to be placed, there has been some debate over the exact placement of security services and other implementation mechanisms. These debates continue as new standards evolve to meet the communication needs of users, local and wide area networking vendors, Internet Service Providers (ISPs), and World Wide Web application designers.

Today we know that data privacy, integrity, and authenticated message delivery are required in a client-server model, where a user's client browser accesses one or more web server's applications. In this model, the inter-network "fabric" of telephone lines, network routers, firewalls, and other network components is seldom under the control of the end user's client software or of the server's application software; and so, data protection against eavesdropping, tampering or message forgery must be placed within the client/server protocols higher in the stack above this inter-network fabric. This ensures that security services are controlled jointly by the client and server, and not by the "fabric".

In a typical Internet architecture, the protocols in this fabric are the Transmission Control and Internet Protocol (TCP/IP) stack plus the protocols below IP in the stack. Protocols below IP include local area network (LAN) protocols or other link protocols such as dial up, or directly connected modems, fiber optic links, or satellite links. The TCP/IP stack provides for the transmission of packets through complex arrangements of local, wide, or metropolitan area or globally connected sets of inter- or intra-networking technology.

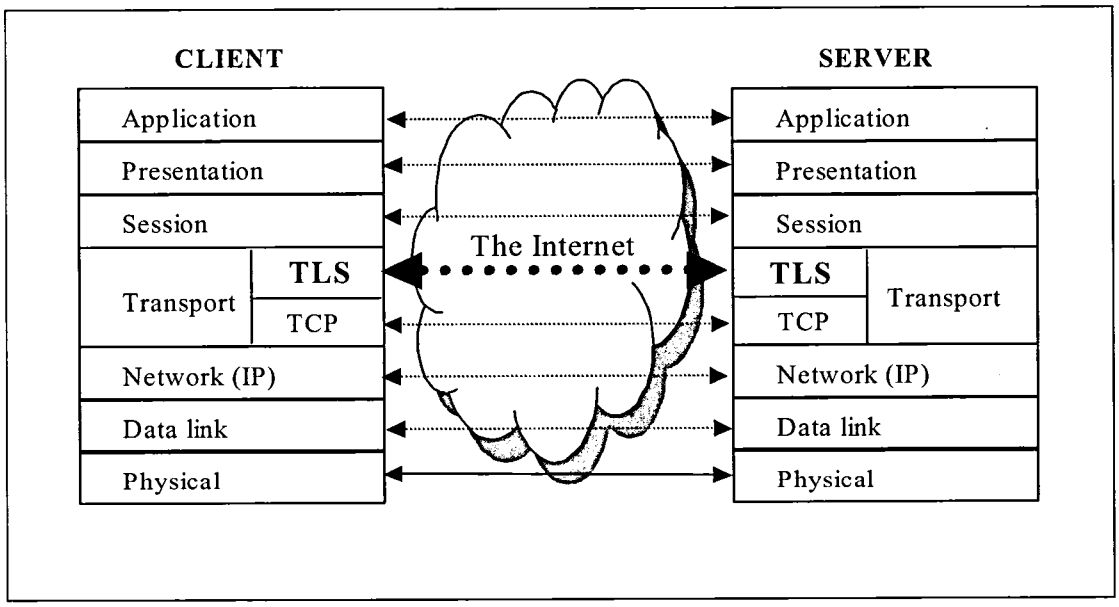

Figure 1: Client/Server Model[2]

Figure 1 depicts this architecture in the context of a client-server model. The arrow between the TLS protocol entities in Figure 1 emphasizes that the TLS entities make no assumptions about the security services provided by the interactions of the protocol entities lower in the stack. All of the security services needed to prevent eavesdropping, tampering or message forgery are provided by the TLS entities themselves. TLS does however rely on the communications functionality of the lower layer protocols to provide end-to-end reliable[3] delivery of data.

It is worth noting that TLS is not the only place in this architectural model where security services could be provisioned. As a general rule, placing services lower in the stack is advantageous because more higher layer entities can utilize the same functionality. However, placing security services lower in the model often requires sharing responsibility for security among many administrative organizations including the local, wide, or metropolitan area network and Internet Service Providers. Many feel that placing security in the transport layer, closer to, or part of, the client's browser and

[2] For simplicity and clarity firewalls and ISPs are not shown in this figure although in most deployed systems, firewalls and ISPs are used for connection to the Internet.

[3] Reliable delivery of data in this context has no security implication. Rather, reliable delivery of data implies that all messages presented to the sending TCP/IP stack are delivered in proper sequence by the receiving TCP/IP stack. These messages may be broken up into packets and fragmented or segmented as they are sent and routed through any arrangement of local, wide area or metropolitan networks. In general, as they are sent and routed through networks, the data are augmented with cyclical redundancy checks or forward error correction techniques to help ensure that the delivered messages are identical to the transmitted messages. Reliable delivery implies that the messages are properly reassembled and presented in correct order (proper sequence) to the peer protocol TLS entity.

server's application, is more appropriate for web-based applications rather than relying on or placing security services in lower levels.

2.1 Security in the Transport Layer

The Netscape Corporation[4] designed a protocol known as the Secure Sockets Layer (SSL) to meet security needs of client browsers and server applications. Version 1 of SSL was never released. Version 2 (SSL 2.0) was released in 1994 but had well-known security vulnerabilities. Version 3 (SSL 3.0) was released in 1995 to address these vulnerabilities. During this timeframe, Microsoft Corporation released a protocol known as Private Communications Technology (PCT), and later released a higher performance protocol known as the Secure Transport Layer Protocol (STLP). PCT and STLP never commanded the market share that SSL 2.0 and SSL 3.0 commanded. The Internet Engineering Task Force (IETF) (a technical working group responsible for developing Internet standards to ensure communications compatibility across different implementations), attempted to resolve, as best it could, security engineering and protocol incompatibility issues between the protocols. The IETF standards track Transport Layer Security Protocol Version 1.0 (TLS 1.0) emerged and was codified by the IETF as [RFC2246]. While TLS 1.0 is based on SSL 3.0, and the differences between them are not dramatic, they are significant enough that TLS 1.0 and SSL 3.0 do not interoperate. However, TLS 1.0 does incorporate a mechanism by which a TLS 1.0 implementation can negotiate to use SSL 3.0 with requesting entities as if TLS were never proposed. However, because SSL 3.0 is not approved for use in the protection of Federal information, (Section 7.1 of [FIPS140Impl]), TLS must be properly configured to ensure that the negotiation and use of SSL 3.0 never occurs when Federal information is to be protected.

These Guidelines attempt to make clear the impact of selecting and using secure web transport protocols for use in protecting sensitive but unclassified U.S. Government information.

2.2 The Security Parts of Transport Layer Security

Both the TLS 1.0 and SSL 3.0 protocol specifications use cryptographic mechanisms to implement the security services that establish and maintain a secure TCP/IP connection. The secure connection prevents eavesdropping, tampering, or message forgery. Implementing data confidentiality with cryptography (encryption) prevents eavesdropping; generating a message authentication code with a secure hash function prevents undetected tampering; and, authenticating clients and servers with public key cryptography-based digital signatures prevents message forgery. In each case – preventing eavesdropping, tampering and forgery – a key or shared secret is required by the cryptographic mechanism. A pseudo-random number generator and a key establishment algorithm provide for the generation and sharing of these secrets.

[4] Commercial company names are used for historical reference purposes only. No product endorsement is intended or implied

The rows in Table 1 identify the key establishment, confidentiality, digital signature and hash mechanisms currently in use today in TLS 1.0 and SSL 3.0. For the purpose of these Guidelines, Table 1 identifies which key establishment, confidentiality, and signature algorithms and which hash functions are Federal Information Processing Standards (FIPS). These FIPS form the basis of the recommendations in these Guidelines.

Table 1: Mapping The Security Parts of TLS to Federal Standards

Mechanism	SSL (3.0)	TLS 1.0	FIPS Reference
Key Establishment	RSA DH-RSA DH-DSS DHE-RSA DHE-DSS DH-Anon Fortezza-KEA	RSA DH-RSA DH-DSS DHE-RSA DHE-DSS DH-Anon	
Confidentiality	IDEA-CBC RC4-128 3DES-EDE-CBC Fortezza-CBC	IDEA-CBC RC4-128 3DES-EDE-CBC Kerberos AES	FIPS 46-3, FIPS 81 FIPS 197
Signature	RSA DSA	RSA DSA EC*	FIPS 186-2 FIPS 186-2 FIPS 186-2
Hash	MD5 SHA-1	MD5 SHA-1	 FIPS 180-2, FIPS 198

Note: DES and all of the "export" algorithms of small key sizes (RC4-40, RC2-CBC-40, DES-40, DHE-DSS-Export and DHE-RSA-Export) have been left out of this table as these are now deprecated.

*An Internet-Draft proposing cipher suites containing Elliptic Curve (EC) algorithms has been introduced in the IETF. See [ECCTLS].

Both TLS 1.0 and SSL 3.0 package one key establishment, confidentiality, signature and hash algorithm into a "cipher suite." Not all combinations from the table work together. Instead, TLS 1.0 and SSL 3.0 implementations contain carefully crafted cipher suites that are registered by the IETF (in the case of TLS 1.0) and the Netscape Corporation (in the case of SSL).[5]

[5] The transport layer security specification assigns a 16-bit (4 hexadecimal digit) number to the defined cipher suite. For example: Ephemeral Diffie-Hellman (DHE) **key agreement**, Digital Signature Algorithm (DSA) **signature**, Triple Data Encryption Standard (3DES) using Encryption-Decryption-Encryption (EDE) and Cipher Block Chaining (CBC) **confidentiality** and the Secure Hash Algorithm (SHA-1) **hash** is assigned the hexadecimal value {0x00, 0x13}. (Note that this suite is not frequently used.) Of special note is the TLS 1.0 requirement that, "In the absence of an application profile standard specifying otherwise, a TLS compliant application MUST implement the cipher suite TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA.", which is represented by this example. However, the current draft of TLS 1.1 mandates TLS_RSA_WITH_3DES_EDE_CBC_SHA, and this suite is more commonly used.

A given implementation of the standard TLS 1.0 or SSL 3.0 protocol may implement one or more cipher suites from the registered set of suites. Finding a common cipher suite between a client and a server is accomplished through a built-in protocol "handshake negotiation" mechanism described in section 3, Negotiating Security Options, on page 13.

2.2.1 Key Establishment[6]

The following key establishment algorithms are used in various cipher suites.

RSA – The sender generates a random session key (pre-master secret) and encrypts it under the recipient's public key. The session key is a symmetric key.

DH (Diffie-Hellman) – The sender and receiver each have key pairs. To compute an agreed-upon session key, the sender combines his private key with the receiver's public key. The receiver combines his private key with the sender's public key.

There are three different types of DH communications, Static (also known as Fixed), Ephemeral, and Anonymous. These are described as follows[7]:

- **Fixed Diffie-Hellman:** This is a Diffie-Hellman key exchange in which the server's certificate[8] contains the Diffie-Hellman public parameters signed by the certificate authority (CA). That is, the public-key certificate contains the Diffie-Hellman public-key parameters. The client provides its Diffie-Hellman public key parameters either in a certificate, if client authentication is required, or in a key exchange message. This method results in a fixed secret key between two peers, based on the Diffie-Hellman calculation using the fixed public keys.

- **Ephemeral Diffie-Hellman:** This technique is used to create ephemeral (temporary, one-time) secret keys. In this case, the Diffie-Hellman public keys are exchanged, and signed using the sender's private RSA or DSS key. The receiver can use the corresponding public key to verify the signature. Certificates are used to authenticate the public keys. This option appears to be the most secure of the three Diffie-Hellman options because it results in a temporary, authenticated key.

---

[6] "Key Establishment" is the process of establishing a shared secret key used for encrypting data exchanged between client and server over a TLS connection. Key establishment is often called "key exchange". In some key establishment schemes (e.g., RSA), the client generates a random key and sends it to the server. In other schemes (e.g., Diffie-Hellman) the server generates some random data, sends the data to the client, the client generates additional random data, combines in with the server's random data, and the resulting key is sent to the server to be used as a secret key. This latter scheme is an example of a "key agreement" type of key establishment because the two sides together agree on a key.

[7] The Diffie-Hellman descriptions are used with the permission of Dr. William Stallings, and appeared in [Stallings98]

[8] For brevity, the term "certificate" is used in these guidelines to mean "X.509 certificate".

- **Anonymous Diffie-Hellman:** The base Diffie-Hellman algorithm is used, with no authentication. That is, each side sends its public Diffie-Hellman parameters to the other, with no authentication. This approach is vulnerable to man-in-the-middle attacks, in which the attacker conducts anonymous Diffie-Hellman exchanges with both parties.

Fortezza-KEA – KEA was the key agreement algorithm used by the Fortezza card supported by the Department of Defense, and KEA was originally classified. It exists in SSL 3.0, but the IETF standards committee did not include it in TLS 1.0.

The rules and protocols for generating and establishing keys, and the handling of those keys throughout their lifecycle, directly affects the level of security achieved. The security and reliability achieved depends on the strength of the key generation process and the protection afforded to those keys. Secret keys and the private key of a public key pair must be protected from disclosure, modification, substitution, and unauthorized deletion.

Some aspects of the key life cycle are addressed by the selection of appropriate cipher suites (i.e., the key establishment algorithm itself) or by the selection of validated modules[9] (e.g., primitives for key generation, storage, and destruction).

In deploying certificates for the support and use of TLS 1.0 and SSL 3.0 it is important to recognize that associated keys must be protected. Users must plan for and deal with many of these generally accepted cryptographic key life-cycle issues:

1) **Generation** of keys is accomplished with the aid of a pseudo-random number generator (PRNG). The protocol handshake sequence "ClientKeyExchange" creates a shared secret of random bits called the pre-master secret.
2) **Establishment** of keys occurs after the "ClientKeyExchange" message used in the handshake sequence creates the shared pre-master secret. This shared secret is expanded using a key derivation function in the client and the server. The key derivation function establishes the individual cryptographic keys that will be used for the various encryptions, authentication and the secure hash functions.[10]
3) **Storage** of keys and the shared secrets used to generate the keys is an issue in transport layer security implementations. Usually, the implementation relies on the capabilities of the operating system to protect this sensitive storage area.
4) **Crypto-periods** of both the session keys and certificate keys are not a transport layer security issue. The established session keys have a lifetime as long as the

[9] Cryptographic modules for Federal Government use must be validated in accordance with [FIPS140-1] or [FIPS140-2].

[10] The key derivation function uses the pre-master key, a label and a seed composed of shared random numbers to develop a master secret from which enough keying material is generated for all the algorithms defined in the cipher suite. This might include up to six keys – an encryption key, a MAC key (see section 2.2.5 for a description of Message Authentication Codes (MACs)) and an initialization vector for both the client and server.

session lasts.[11] Any keys used in the optional X.509 certificates that identify clients and servers have a crypto period that is determined by the policy of the certificate issuing authority.

5) **Recovery** of session keys is not a transport layer security issue. There is no requirement for a key recovery mechanism in the transport layer security protocol because the shared session key is ephemeral, a new session can be easily established with a new ephemeral key.[12]

6) **Destruction** of all of the state variables including the session keys occurs when the session ends. The protocol implementation relies on the operating system to insure that there is no reuse of storage areas, which may contain sensitive information.

2.2.2 Confidentiality

The following symmetric encryption algorithms are used in various cipher suites to provide confidentiality:

IDEA[13] – IDEA is a block cipher that operates on 64 bit plaintext blocks. The key is 128 bits long. The same algorithm is used for encryption and decryption. IDEA is not FIPS-approved.

RC4 – RC4 is a stream cipher that uses a variable length key of anywhere between 8 and 2048 bits long. RC4 is not FIPS-approved.

3DES-EDE – The Data Encryption Standard (DES) is the most widely used symmetric block cipher. It uses 64 bit blocks and a 56-bit key. Triple DES (also known as 3DES) super-encrypts by running the data through the DES algorithm 3 times with different keys. The first time it **E**ncrypts with key 1, the second time it **D**ecrypts with key 2 and the third time it **E**ncrypts again with key 3; hence the acronym 3DES-EDE. 3DES-EDE is FIPS-approved.

AES – The Advanced Encryption Standard is a FIPS-approved symmetric block cipher encryption algorithm that may be used by U.S. Government organizations (and others) to protect sensitive but unclassified information. AES uses 128, 192, or 256 bit keys, however cipher suites have only been defined for 128 and 256 bit keys to reduce the over proliferation of cipher suites. The block

---

[11] The message sequence space for SSL 3.0 and TLS 1.0 is 64 bits long; so, there is a theoretical limit of 2^{64} messages per session. If this limit is reached the connection must close as the sequence number cannot repeat or recycle. Furthermore, there is no mechanism to "rekey" the session. In effect this limits the crypto period of the session key.

[12] Servers and clients may (and often do) cache the master secret (but not the session key) to reduce the significant overhead in session resumption. After some reasonable timeout period, the master secret should be destroyed on both the server and the client.

[13] Cipher Block Chaining (CBC) is one of four DES modes of operation and can be used with all of the symmetric block ciphers listed here (i.e., IDEA, 3DES-EDE, and AES). CBC chains together the ciphertext of one block with the plaintext of the next block. Additional information on CBC can be found in [FIPS 46-3].

size in AES is 128 bits. The AES algorithm [FIPS197] is designed to replace DES and 3DES. AES is FIPS-approved.

Note that RC4 is currently the most commonly used confidentiality algorithm in SSL/TLS. However, RC4 is not FIPS-approved.

2.2.3 Signature

The following digital signature algorithms are used in various cipher suites:

RSA – To sign a message with the RSA algorithm the signatory computes a message digest or hash of the message and encrypts it with the private key. To verify an RSA signed message, the verifier decrypts the message digest with the signatory's public key and compares it with a locally computed hash of the original message. If the decrypted hash matches the locally calculated hash, the signature is valid. The use of RSA for digital signatures requires encryption and decryption.

DSA – To sign a message with the DSA algorithm, the signatory computes a signature using the SHA-1 hash algorithm and the signatory's private key. To verify a DSA signature, the verifier performs a computation using the message hash, the signatory's signature and the public key. The DSA verifier returns a yes or no rather than a decrypted hash as in the RSA signature. There is no encryption or decryption performed.

2.2.4 Hash

Hash algorithms used for cryptography (one-way hashes) have the property that it is computationally infeasible to find two messages that hash to the same value. When a message of any length is input to an algorithm, the result is an output called a message digest. The message digests range in length, depending on the algorithm used, however for use within Government environments four lengths have been specified; 160 bits, 256 bits, 384 bits, and 512 bits. Hash algorithms are identified as being secure if for a given algorithm, it is computationally infeasible that a message corresponding to a given message digest will be found, as well as finding two different messages producing the same message digest. Any change to a message will, with a very high probability, result in a very different message digest. This will cause a verification failure when the hash algorithm is used in conjunction with a digital signature algorithm or a keyed-hash message authentication algorithm.

The following hash algorithms are used in various cipher suites:

SHA–1: An algorithm for computing a condensed representation of a message or a data file. When a message of any length less than 2^{64} bits is input, SHA-1 produces a 160-bit output called a message digest. (The 2^{64} bit limit is caused by a 64-bit field size for a length descriptor used by SHA-1.)

MD5: When a message of any length is input, MD5 produces a 128-bit message digest. MD5 is not a FIPS-approved hash function, but it is in common use.

Note: Currently SHA-256, SHA-384, and SHA-512 are not featured in standard SSL or TLS cipher suites. It is expected that these variants of the secure hash algorithm will be added to TLS.

2.2.5 MAC

Message authentication (and integrity checking) is often achieved through the construction of a message authentication code (MAC), which is a small amount of additional data that is sent along with a message. To use a MAC, the sender and recipient must have a shared secret key known to no one else. Before sending a message, the sender computes the MAC as a function of the message and the key. Then the sender sends the message along with the computed MAC. The recipient computes a new MAC for the received message using the message itself, the secret key, and the same function as the sender. If the newly computed MAC is the same as the MAC received with the message, the recipient can be sure that the message has not been modified in transit, and that the sender knows the same secret key. That knowledge of the secret key assures the recipient of the sender's identity.

Cryptographic hash functions (with properties as described in 2.2.4 above) are often used as the function used to create MACs. In a simple scheme the secret key is simply prepended to the message, and the hash of this concatenation is used as the MAC. Message authentication codes generated using hash functions are known as HMACs and are used in TLS, SSL, and other common security mechanisms. For a detailed description of HMAC operation see [RFC2104] and [FIPS198].

A number of operations in the TLS protocol require an HMAC for authentication and integrity. Forging these MACs is infeasible without knowledge of the MAC secret key thus providing the necessary level of trust between communicating parties. Although HMACs can be used with a variety of different hash algorithms, TLS only specifies use of MD5 and SHA-1. To provide additional security for handshakes within the protocol, TLS uses both MD5 and SHA-1 in conjunction with each other to produce the HMAC.

3 Negotiating Security Options

Figure 2 provides details of the transport layer security protocol entities. The entities have been developed to allow control information to flow between the client and the server. Three types of control messages are exchanged: "Handshake", "ChangeCipherSpec, and "Alert". In addition, the entities exchange application data between the client and server protected by the security services provisioned by the negotiated cipher suite. These security services are negotiated and established with the Handshake.

The "Handshake" consists of a series of message exchanges between the client and the server. This series of messages is used to establish the TLS/SSL environment, including a set of negotiated security algorithms. The ChangeCipherSpec message is used to inform the other side to begin using the negotiated security services by changing to the cipher suite agreed to during the Handshake. All messages sent after ChangeCipherSpec message are encrypted using the just negotiated cipher suite.

The Alert provides a way to signal special security channel events and errors, as well as asynchronous events that may occur during a TLS/SSL session. In addition, an alert message is used to terminate a session.

Specific details of the Handshake, ChangeCipherSpec and Alert message exchanges are beyond the scope of these Guidelines; but understanding that the Handshake negotiates a cipher suite is important to understanding how to configure and use Transport Layer Security.

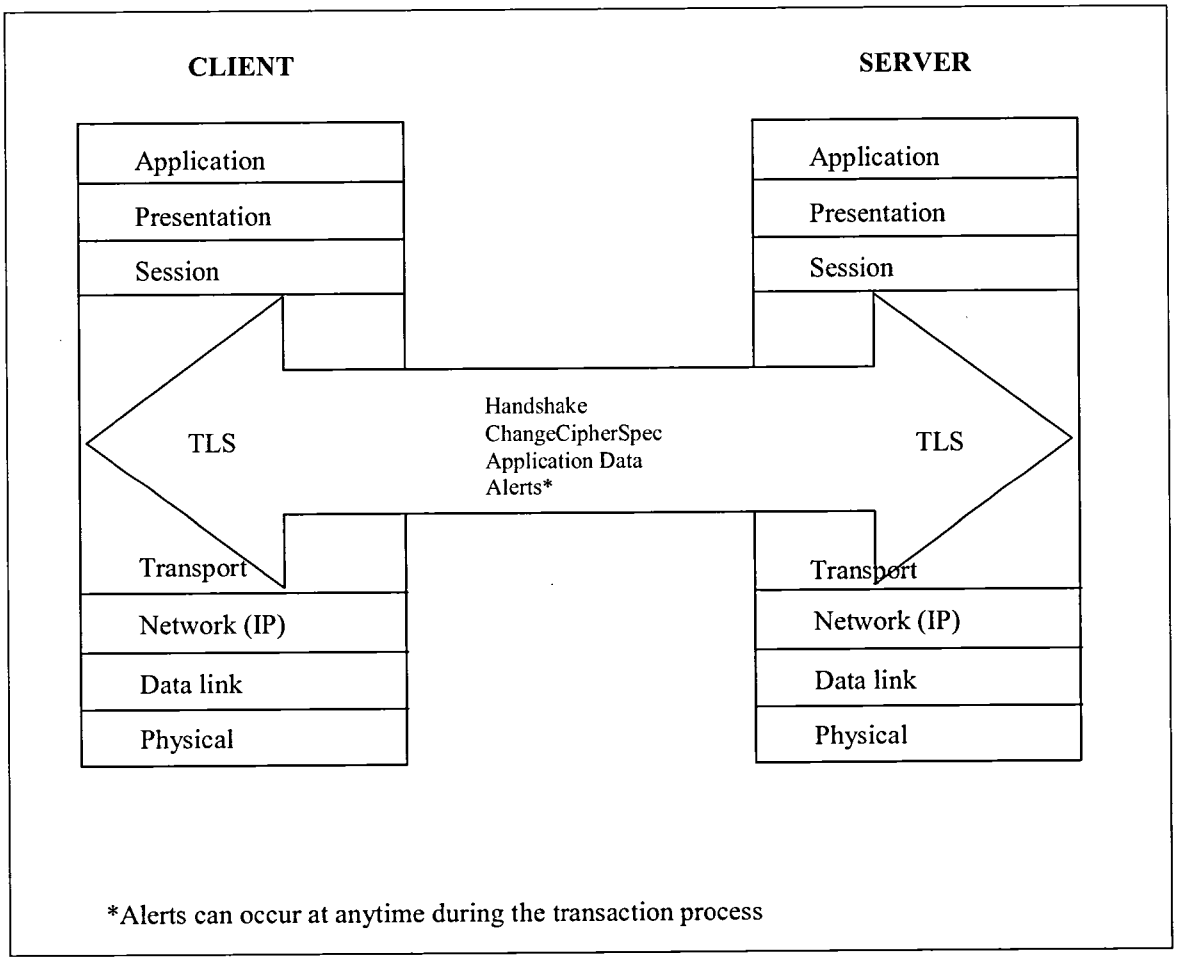

Figure 2: The Transport Layer Security Protocol Entity

The Transport Layer Security handshake protocol establishes a secure channel inside of a TCP/IP connection before passing any data from the application.

- The handshake protocol initializes both the client and server to use optional cryptographic capabilities by negotiating a cipher suite of algorithms and functions including key establishment, digital signature, confidentiality and integrity algorithms with their respective key sizes, and hash functions. This negotiation begins with the "ClientHello" message and continues with the "ServerHello" message.
- The handshake protocol may exchange public key digitally signed X.509 certificates[14] to optionally authenticate the server to the client and vice versa. In most cases, the server presents a certificate to the client, but the client does not present a certificate to the server. However, TLS and SSL allow for certificates to be presented by a server, by a client, by both, or by neither in negotiating a

[14] The use of X.509 certificates is fundamental to TLS/SSL, as well as other PKI-enabled services. For a comprehensive explanation of X.509 certificates see, for example, [Adams99] or [Housley01].

session. The important point to note is that presenting a valid X.509 certificate and proving possession of the private key authenticates the presenter to the recipient.

- Using the negotiated key establishment algorithm, the handshake protocol exchanges random data for developing keying material to be used by the cryptographic algorithms.

When all the security parameters are in place (i.e., after the Handshake), the "ChangeCipherSpec" message engages the negotiated cryptographic capabilities for application data exchanges.[15]

After the handshake protocol completes, application data may be exchanged between the client and server. This data is encrypted and integrity checked using the security services that were negotiated during the handshake.

Alerts are used to signal asynchronous/exceptional events during a TLS/SSL session. For example, an alert can be used to signal "decrypt_error" or "access_denied". Some alerts are used for warning, and others are considered "fatal" and can lead to immediate termination of the session. A "close_notify" alert is used to signal normal termination of a session. Like all other messages, after the handshake protocol is completed, alert messages are encrypted and optionally compressed.

3.1 The Cipher Suite

The negotiated algorithm identifiers are referred to collectively as the "Cipher Suite". Cipher suites are identified in human readable form using a mnemonic code. Each cipher suite includes a code for the protocol – either SSL or TLS. The next mnemonic identifies the key establishment algorithm and digital signature algorithm followed by the word "WITH" followed by the confidentiality algorithm followed by the hash function. An example is:

TLS_DHE_DSS_WITH_3DES-EDE-CBC_SHA

This cipher suite is for:

- A Transport Layer Security Version 1.0 protocol, TLS,
- The Ephemeral Diffie-Hellman key agreement, DHE,
- The Digital Signature Standard, DSS (which implies the Digital Signature Algorithm, DSA),
- The Triple Data Encryption Standard's Encrypt-Decrypt-Encrypt option in Cipher Block Chaining mode, 3DES-EDE-CBC, and
- The Secure Hash Algorithm, SHA-1 (used to compute a HMAC).

[15] Although not specifically related to security, the handshake also optionally negotiates a compression algorithm for the application data exchanges.

Both TLS (1.0) and SSL (3.0) specify registered index numbers for the cipher suites. When negotiating a cipher suite, the client (which always initiates TLS/SSL sessions) sends a handshake message with a list of cipher suite indexes it will accept. The server chooses from the list and sends a handshake message back indicating which cipher suite it will accept. Although the client orders the list with the "strongest" cipher suites listed first, the server may choose **ANY** of the cipher suites proposed by the client. Therefore there is **NO** guarantee that the negotiation will settle on the strongest suite in common. If no cipher suites are in common the connection is aborted.

When the negotiated options are complete, the client sends a "ChangeCipherSpec" and a "Finished" message. In response, the server also sends a "ChangeCipherSpec" and a "Finished" message to place the communication channel in a secure mode to protect transmission of the client's browser and server's application data.

3.2 Data Integrity of the Handshake

Data integrity is maintained throughout the handshake process and finally completed with the sending of the "Finished" message. A "Finished" message is always sent immediately after the "ChangeCipherSpec" message to verify the key exchange and authentication processes were successful. With successful exchanges of this message the client and server verify that the entire handshake has not been modified. This verification is possible because each side sends a hash of the concatenated handshake messages to the other side, which compares it to the same result computed locally. If the hash values differ, the handshake has been modified and the connection is aborted. If the hash values are the same, there is high assurance that the entire handshake has cryptographic integrity – nothing was modified, added or deleted.

4 Recommendations

This section presents criteria for developing specific recommendations when selecting, installing and using a transport layer security.

4.1 Selection Criteria

When implementing transport layer security mechanisms (usually web servers and browsers), there are several important aspects to consider such as whether the implementation:

- **Is standards-based** – The interaction between components in a transport layer security mechanism should be a well-defined communication protocol with no deviations. Additionally, FIPS-approved algorithms for authentication, encryption, and the generation of message digests should be used in all implementations.
- **Supports interoperability** – Any implementation should promote interoperability among components. The selection of a particular server solution should not prevent the use of any standards-based client or vise versa.
- **Includes evaluated products** – Key components of the implementation should be independently evaluated against known standards (e.g., cryptographic modules validated against FIPS 140-1 or 140-2).
- **Select important features** – The implementation should include those features that users consider most important to their operating environment.
- **Is Open Source** – The implementation should be an open source solution to help prevent being locked into a proprietary implementation that may not support interoperability or identified standards in the future.

4.2 Protocol Selection

Due to the vulnerabilities inherent in SSL 2.0, the only reasonable protocols to consider for deploying transport layer security are SSL 3.0 or TLS 1.0. However, using the criteria discussed in 4.1, TLS 1.0 is the only acceptable alternative. It is an IETF "standard" that incorporates FIPS-approved algorithms while SSL 3.0 is not standards-based, resulting in several competing incompatible SSL variants. Because SSL 3.0 uses some non-FIPS-approved cryptographic algorithms, it is not approved for use in the protection of Federal information [FIPS140Impl]. Likewise there exist incompatibilities between SSL 3.0 and TLS 1.0 implementations that prevent the integrated use of both TLS 1.0 and SSL 3.0 components, such as:

- SSL 3.0 allows a party to start sending application data as soon as that party's "Finished" message is sent. TLS requires a "wait" until the other "Finished" message is received.

- SSL 3.0 "ClientKeyExchange" contains two bytes representing the SSL version number. Some implementations use this value as a negotiation. Use of these two bytes as a negotiable value is incompatible with TLS 1.0.
- The key derivation from the pre-master secret is different. (SSL uses a key derivation model in which half the master secret is generated using only MD5, which is not a FIPS-approved hash algorithm.).
- SSL 3.0 omits the length bytes when RSA is used for sending the encrypted pre-master secret.
- Alert messages produce different results.

Although the selection of TLS may cause legacy problems with clients that are incapable of supporting TLS, almost all currently used clients (e.g., Internet Explorer, Netscape, Mozilla) support TLS.

4.3 Cipher Suite Selection

The last column in **Table 1: "Mapping the Security Parts of TLS to Federal Standards"** on page 7, lists the Federal Information Processing Standards that are applicable for use in various components of the negotiated cipher suites. Table 2 and Table 3 in Section 5 provide recommendations for the cipher suites that should be offered based, on the information presented below.

It is important to note that RFC 2246 specifies all TLS compliant applications implement the TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA cipher suite. This document does not supplant this requirement, but instead goes beyond RFC2246 and specifies only the following cipher suites may be used.

Key Establishment: For maximum security, RSA or DSA authentication with ephemeral Diffie-Hellman key agreement is recommended (e.g., AES might be TLS_DHE_RSA_WITH_AES-128_CBC_SHA and TLS_DHE_DSS_WITH_AES-256_CBC_SHA.).

These cipher suites provide perfect forward secrecy[16], and can support large keys for long-term confidentiality.

The Federal Government has not finalized a specification for FIPS-approved or NIST-Recommended algorithms for establishment of cryptographic keys. Note, however that a NIST Recommendation is currently under development. See [KeyMgmt03]. Upon completion, only recommended algorithms shall be used.

Confidentiality: Since TLS 1.0 is the security protocol of choice within the transport layer; only 3DES-EDE-CBC or AES should be offered during the handshake. Clients should never offer the deprecated 40 or 56-bit suites. While encryption is optional,

[16] Perfect forward secrecy is the condition in which the compromise of a session key or long-term private key after a given session does not cause the compromise of any earlier session key.

anonymous cipher suites are not allowed. For use in government agency to government agency communication, only FIPS-approved algorithms shall be used.[17]

Signature: The use of 1024 bit (key size) RSA/DSA server X.509 certificates is acceptable until the year 2010. If the client identity must be authenticated after the year 2010, key sizes larger than 1024 bits are needed.

Note that although RSA keys should have a key size of at least 1024 bits, many browsers and SSL/TLS servers have no mechanism to allow users to explicitly specify a minimum acceptable key size. However, by setting the symmetric key size to 128 bits (allowed by many browsers and servers) then an RSA key size of 1024 is often implicit in these browsers.

Hash: TLS offers two options for a cryptographic hash algorithm. These are SHA-1 and MD5. While SHA-1 is a FIPS-approved cryptographic hash algorithm, MD5 is not and thus should not be offered as a selection.

Note: The choice of a hash function within a cipher suite selection affects only the HMAC used to protect payload traffic. Other uses of hashes in key derivation or signatures are directly affected by cipher suite choice: RSA client authentication signature always uses both SHA-1 and MD5 while DSA client signatures only uses SHA-1.

[17] RC4 is acceptable for use on Government "Client" systems in very limited circumstances where secure information is to be transferred between Government systems and non-government servers, and 3DES or better (e.g., AES) is not supported by the server. For example, many vendor web sites providing supplies to the government support nothing stronger than RC4, and credit card information must be conveyed and secured to order supplies. In such cases risk is limited to exposure of government credit card information, and agencies may wish to take this risk to expedite ordering of supplies. RC4 should never be used on Government "Server" systems where government owned/generated data is to be made available in a secure manner to "client" systems.

5 Guidance

This section offers guidance to system administrators responsible for procuring, installing and maintaining implementations of transport layer security protocols.

Implementation Selection/Procurement – Among the responsibilities of any implementer or vendor of security functionality are:

- To provide quality random numbers for key generation,
- To protect the keying material and its storage,
- To properly implement and test key establishment, encryption, and signature algorithms and hash functions.

The procurement authority should ensure that the implementation meets a minimum set of universally accepted tests. Guidance for buying security products is provided in [SP800-23], Guidelines to Federal Organizations on Security Assurance and Acquisition/Use of Tested/Evaluated Products and [SP800-36], Guide to Selecting Information Technology Security Products.

Installation – Installation guidance usually involves following the vendor's general guidelines for configuration of the TLS or SSL protocol and following local (either to the server or to the client) configuration policy for selecting the optional protocol security services. For example, a client's local policy might state that server authentication is required. The system administrator would follow the vendor's prescribed methods for enabling client/server authentication.

The following security services should be configured and provided by the TLS implementation:

- Confidentiality,
- Data integrity,
- Peer entity authentication for clients and servers.

Appropriate cipher suites must also be selected and their priority ordered from strongest to weakest acceptable.

When peer entity authentication is selected, special attention should be paid to the public key infrastructure constructs used by TLS and SSL. Here, X.509 certificates are used. For example, for a client to be authenticated to a server, the authentication process requires that there must be a valid certification path that starts with one of the client's trust roots[18]. Part of the installation process is to ensure that all required trusted roots are present in the client and server implementations.

---

[18] A "trust root" (also known as a "trust anchor") is a X.509 certificate issued by (and signed by) a trusted authority. Before accepting a X.509 certificate (e.g., one presented by a TLS server) as valid, the user (known as a "relying party") must check to see that the certificate is signed by a trust root, or signed by an intermediate trusted authority whose certificate is signed by a trust root. There may be multiple intermediate trusted authorities, but the user must be able to find a chain of certificates that can traced back to a trust root before the user can rely upon a X.509 certificate. For a comprehensive explanation of X.509 certificates and trust see, for example, [Adams99] or [Housley01].

Note that for most common usage TLS and SSL are used to authenticate servers and not to authenticate clients. For merchandising implementations, for example, clients need to validate that they are dealing with an authenticated merchant before entering credit card information. The merchant only cares that a valid credit card is presented. (He does this by communicating with the credit card issuer. The details of that validation are beyond the scope of these guidelines.) The merchant does not need to authenticate the buyer.

Although outside of the installation process for the TLS implementation, another consideration is the adequate storage and protection of the client's and/or server's private key within a secure cryptographic module or token. This will prevent the masquerading of one party within the connection.

Maintenance – Once the server or client implementation is installed and operational, maintenance of the product generally follows local policies and operating procedures. For example, the site system administrator may be required to check for product updates and patches and install as needed. Within the local operating procedures, provisions need to be made for checking for and obtaining updated certificate revocation lists (CRLs) or using any on-line validation mechanisms available from the Certification Authority[19].

5.1 Considerations for Selecting TLS Client Implementations

Clients play a limited, but crucial role in the overall security posture. The client negotiates three parameters: the protocol version, the cipher suite, and the compression algorithm. These items are presented in the "ClientHello" message and form the basis for the server to negotiate the strongest possible security options.

The ClientHello message is the first message to be sent as the client establishes a TLS connection to the server. These messages allow the client to stay connected, re-establish an existing session, or to establish several independent secure sessions without repeating the full handshake procedure.

The client version field within the ClientHello message represents the protocol version that client supports. This field should contain the highest version number the client is prepared to support. For implementations that support TLS this value is: major=3, minor=1 (which represents 3.1, and hence TLS). All non-TLS implementations should use major=3, minor=0 for SSLv3. This designation does not limit the implementation to the identified protocol version. For example, if a client wishes to use only TLS, the client must connect to the server and is responsible for terminating the connection if the server selects any other protocol. Under no circumstances should a client use any protocol less than SSLv3[20]. For the most secure protection of data, only use clients that support TLS and that can disable all versions of SSL.

---

[19] Certificate authorities are trusted authorities that issue X.509 certificates to end users and to intermediate certificate authorities. When a certificate authority determines that a certificate has been compromised or should be considered invalid for further use, it adds the serial number of that certificate to its list of revoked certificates (i.e., to its CRL). Certificate authorities periodically distribute updated CRLs. Users (relying parties) should check the CRL before using an otherwise valid X.509 certificate to ensure that it has not been revoked.

[20] Only TLS can be used for the protection of Federal data. There may be instances where Federal users need access to non-Federal sites that do not support TLS. In these instances, Federal managers may allow the use of SSLv3 to transfer non-Federal data. However, the SSLv3 should be used in only in limited, low risk situations, and non-Federal sites should be encouraged to support TLS.

To ensure the security of the connection, client implementations should support the cipher suites presented in Table 2. This table presents the cipher suites listed in order of descending security strength. Unfortunately the order of these cipher suites is ignored by the server, which will select any cipher suite that it prefers from those offered by the client.

Table 2: Recommended Client Cipher Suites[21]

Cipher Suite	Authent-ication	Key Establishment	Encryption	Digest
TLS_DHE_DSS_WITH_AES_256_CBC_SHA	DSS	DHE	AES_256_CBC	SHA-1
TLS_DHE_RSA_WITH_AES_256_CBC_SHA	RSA	DHE	AES_256_CBC	SHA-1
TLS_RSA_WITH_AES_256_CBC_SHA	RSA	RSA	AES_256_CBC	SHA-1
TLS_DH_DSS_WITH_AES_256_CBC_SHA	DSS	DH	AES_256_CBC	SHA-1
TLS_DH_RSA_WITH_AES_256_CBC_SHA	RSA	DH	AES_256_CBC	SHA-1
TLS_DHE_DSS_WITH_AES_128_CBC_SHA	DSS	DHE	AES_128_CBC	SHA-1
TLS_DHE_RSA_WITH_AES_128_CBC_SHA	RSA	DHE	AES_128_CBC	SHA-1
TLS_RSA_WITH_AES_128_CBC_SHA	RSA	RSA	AES_128_CBC	SHA-1
TLS_DH_DSS_WITH_AES_128_CBC_SHA	DSS	DH	AES_128_CBC	SHA-1
TLS_DH_RSA_WITH_AES_128_CBC_SHA	RSA	DH	AES_128_CBC	SHA-1
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DSS	DHE	3DES_EDE_CBC	SHA-1
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA	RSA	DHE	3DES_EDE_CBC	SHA-1
TLS_RSA_WITH_3DES_EDE_CBC_SHA	RSA	RSA	3DES_EDE_CBC	SHA-1
TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA	DSS	DH	3DES_EDE_CBC	SHA-1
TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA	RSA	DH	3DES_EDE_CBC	SHA-1
TLS_RSA_WITH_RC4_128_SHA[22]	RSA	RSA	RC4_128	SHA-1

At this time, compression options have not been defined for either TLS[23] or any version of SSL. However, OpenSSL and some proprietary implementations support private compression algorithms. Care should be given to ensure that these proprietary and/or private algorithms, if implemented, do not weaken the security posture of the protocol. Also, many implementations do not support compression, so compression may not be possible during a TLS/SSL session, even if desirable.

5.2 Server Considerations

Cipher Suites

Although the client may present the cipher suites that it prefers in order of descending preference, the server generally does not defer to the client's preferred cipher suite. The server may, at its choosing, select a common cipher suite that it prefers. The following

---

[21] TLS has standardized AES cipher suites, however, it is expected that these cipher suites are not yet widely supported by commercial products. As products begin to provide support for AES, client implementations should support AES cipher suites as the highest priority cipher suites.

[22] RC4 is not a FIPS-approved cryptographic algorithm. For this reason, cipher suites with RC4 should be offered only when communicating with non-government entities in limited, low risk situations for the transfer of non-Federal data when a FIPS-approved encryption algorithm is not supported. Normally this cipher suite should not be offered.

[23] An Internet Draft has been proposed to define compression methods for TLS [Hollenbeck04].

table (Table 3) represents the cipher suites that a TLS server implementation should support. This table presents the cipher suites in order of descending preference.

Table 3: Recommended Server Cipher Suites[24]

Cipher Suite	Auth	Key Establishment	Encryption	Digest
TLS_DHE_DSS_WITH_AES_256_CBC_SHA	DSS	DHE	AES_256_CBC	SHA-1
TLS_DHE_RSA_WITH_AES_256_CBC_SHA	RSA	DHE	AES_256_CBC	SHA-1
TLS_RSA_WITH_AES_256_CBC_SHA	RSA	RSA	AES_256_CBC	SHA-1
TLS_DH_DSS_WITH_AES_256_CBC_SHA	DSS	DH	AES_256_CBC	SHA-1
TLS_DH_RSA_WITH_AES_256_CBC_SHA	RSA	DH	AES_256_CBC	SHA-1
TLS_DHE_DSS_WITH_AES_128_CBC_SHA	DSS	DHE	AES_128_CBC	SHA-1
TLS_DHE_RSA_WITH_AES_128_CBC_SHA	RSA	DHE	AES_128_CBC	SHA-1
TLS_RSA_WITH_AES_128_CBC_SHA	RSA	RSA	AES_128_CBC	SHA-1
TLS_DH_DSS_WITH_AES_128_CBC_SHA	DSS	DH	AES_128_CBC	SHA-1
TLS_DH_RSA_WITH_AES_128_CBC_SHA	RSA	DH	AES_128_CBC	SHA-1
TLS_DHE_DSS_WITH_AES_256_CBC_SHA	DSS	DHE	AES_256_CBC	SHA-1
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DSS	DHE	3DES_EDE_CBC	SHA-1
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA	RSA	DHE	3DES_EDE_CBC	SHA-1
TLS_RSA_WITH_3DES_EDE_CBC_SHA	RSA	RSA	3DES_EDE_CBC	SHA-1
TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA	DSS	DH	3DES_EDE_CBC	SHA-1
TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA	RSA	DH	3DES_EDE_CBC	SHA-1

Note that all server certificates with RSA keys should have a key length of at least 1024 bits.

Client Authentication

Client authentication when required is accomplished during the handshake. The server initiates client authentication by requesting the client's certificate and providing guidance as to the types of certificates and algorithms the server will accept. The exchange is completed when the client responds with its certificate and a signed hash of the original handshake to prove possession of the corresponding private key.

Although the protocol allows the server to continue the connection using another authentication mechanism (e.g., username and password – not as strong but often used) if a client does not have a suitable certificate, for strong authentication or forgery prevention, server implementations should not allow the connection to be established. In the event that a client does not have a certificate or an acceptable certificate, the server should terminate that connection with a fatal "handshake failure" alert.

The TLS installations almost always require that servers use a certificate and their private key to authenticate themselves to clients. TLS client authentication is optional, and requires that the client have a suitable certificate, issued by a certification authority accepted by the server.

---

[24] TLS has standardized AES cipher suites [RFC3268], however, it is expected that these cipher suites are not yet widely supported by commercial products. As products begin to provide support for AES, server implementations should support AES cipher suites as the highest priority cipher suites.

In the most common use of TLS, only the server is authenticated through the TLS protocol itself. The client is assured that:

- the server has a certificate issued by a CA accepted by the client (as a practical matter this means a certificate issued by a CA with a self-signed root certificate that is in the in the client's trusted certificate store);
- the server controls the private key corresponding to the public key in the server's certificate, and;
- that the server's network address is consistent with the address in the server's certificate.

Browser clients typically display a "lock" symbol to inform the client that a secure, "https" session is in effect. The TLS session is frequently used primarily to protect a user password from eavesdroppers, and, in these cases, the authentication of the client to the server application depends entirely on the password, not the TLS protocol. Unless clients carefully manage the contents of the trusted certificate store and check the URI displayed in their browser's windows, it may be possible for a sophisticated "man-in-the-middle" attacker to successfully impersonate a web server and intercept protected data, including passwords.

Where strong cryptographic client authentication is required, the server should use the TLS protocol client authentication option to request a client certificate and use that certificate to cryptographically authenticate the client. The server can also provide the client with a list of the CAs it recognizes. In a successful client-authenticated TLS session both parties are assured that:

- the other party has a certificate issued by an acceptable CA (as a practical matter this means a certificate issued by a CA with a self-signed root certificate (possibly through intermediate CAs) that is in a trusted certificate store), and;
- the other party controls the private key corresponding to the public key in its certificate

In addition, the client knows that the server's network address is consistent with the address in the server's certificate.

When client authentication is used, both parties are strongly authenticated, and all data transferred in the TLS session is protected and bound to the authentication by symmetric keys generated in the authentication process. Unless clients carefully manage the contents of the trusted certificate store and check the URI displayed in their browser's windows, it may be possible for a sophisticated "man-in-the-middle" attacker to successfully impersonate a web server and intercept protected data. However, with client authenticated TLS, it is not possible for the attacker to learn the client's password or private key.[25]

[25] When TLS client authentication is used, there is no client password used, and the private key is never transmitted nor divulged.

Session Resumption

During the initial handshake between the client and server, the server generates a session id and passes this value to the client in the "ServerHello" message. The session id (along with the key material and cipher suite) is stored for later use after completion of the handshake. If the server is willing to resume a session at the request of a client (resubmission of "ClientHello"), the server responds with the original session id and cipher suite in the "ServerHello" message. In the event the server is unwilling to resume the session, the server generates and responds with a new session id.

Typical server implementations are agreeable to resuming a previous session. This is a secure mode of operation as the session keys (along with the pre-master secret and master secret) are known only to the client and server, and are coupled with the initial client authentication to provide the necessary security. If there is a requirement to authenticate each client as they initiate a connection session, the server should be configured to ignore requests to resume a session, and generate a new session id, which forces the entire handshake procedure (including client authentication) to proceed.

5.3 Generation of Random Numbers

Of particular concern to both the client and server is the generation of quality random numbers. Random numbers are used for the generation of keys, and for the generation of any random number that is needed to complete cryptographic exchanges. As such, it is important to strive for the following principles when generating random numbers:

- All possible outputs should occur with equal probability, and a series of outputs should appear to conform to a uniform distribution.
- Given a sequence of output bits, it should not be feasible to compute or predict any other (past or future) output bit.

Random numbers can be obtained using either a non-deterministic random bit generator (NRBG) or a deterministic (pseudorandom) random bit generator (DRBG). NRBGs and DRBGs produce bit strings from which random numbers can be determined. Both types of generators present implementation problems. DRBGs must be properly seeded with random data that should normally be kept secret, and the generator must provide the capability of generating a very large stream of bits without repeating. NRBGs rely on some unpredictable, physical source of randomness that is outside human control to produce random output. Typically, an NRBG may not produce random bits quickly enough. A simple solution to the problem of producing random numbers quickly is to use an NRBG to seed a DRBG. Care must be taken in the design and use of either type of generator to ensure that the requirements for randomness are met. Therefore, random numbers should be generated in modules that are validated under the Cryptographic Module Validation Program (CMVP)[26].

---

[26] The Cryptographic Module Validation Program (CMVP) is a joint venture of NIST and the Communications Security Establishment (CSE) of the Government of Canada. The CMVP validates commercial products for conformance to FIPS 140-1 ([FIPS140-1]) and FIPS 140-2 ([FIPS140-2]), allowing vendors to build to a common standard and utilize one common validation process. Additional information on the CMVP is available at Hhttp://cs-www.ncsl.nist.gov/cryptval/cmvp.htmH.

Note: This is an effort underway within the American National Standards Institute (ANSI) to develop a Random Number Generator standard. NIST plans are to use this as a basis for a Federal standard.

5.4 Operational Considerations

5.4.1 Implementation Considerations

Sections 5.1 and 5.2 of this document provide recommendations for the cipher suites that the clients and browsers should implement for transport layer security within the TLS protocol. System administrators need to fully understand the ramifications of selecting cipher suites and configuring applications to support only those cipher suites. Through current NIST research into products supporting TLS, it was determined that:

- The set of allowed cipher suites for most browsers includes, by default, RC4 for encryption with a 40 bit key,
- There was a limited choice of cipher suites for browsers and servers, and cipher suites with RC4 were typically chosen first,
- Most server implementations do not allow the server administrator to specify preference order. The only way to ensure that a server uses 3DES for encryption, was to configure the server to not implement cipher suites with RC4, and
- On many systems the selection of a cryptographic algorithm was system-wide and not application specific (e.g., disabling an algorithm for one application would disable that algorithm for all applications on Microsoft Windows systems using Microsoft's Cryptographic API).

Of equal importance is the need to specify the key lengths used in the cipher suites (for both clients and servers). Currently, most browsers do not allow for user specification of key lengths, but those utilizing OpenSSL libraries can select either 512 or 1024 bit RSA key sizes. The use of RSA/DSA server X.509 certificates with a minimum 1024 bit key size is acceptable until the year 2010. If the server identity must be authenticated after the year 2010, key sizes larger than 1024 bits are needed.

5.4.2 Additional Operational Concerns

The Hypertext Transfer Protocol (HTTP) is an extremely flexible protocol that allows for many uses and implementations. This flexibility, however, also introduces vulnerabilities that are not and cannot be mitigated solely by Transport Layer Security. Given the ease at which connections to a known server can be hijacked, it is incumbent upon the client to not only check all data received but also verify the pathway of the message and the message's integrity. This includes verifying the server's identity presented in the server's certificate at the time the connection is established.

Likewise, both the server and the client should not base authentication decisions solely upon the Transport Layer Security's mechanism for determining possession of the private

key that corresponds to the exchanged certificate. Rather, the decision should also consider whether or not the certificate is valid or has been revoked.

6. References

The following list of documents, publications, and organizations provide a wide variety of information on varying aspects of Transport Layer Security.

[Adams99] Adams, C. and Lloyd, S., *Understanding PKI: Concepts, Standard, and Deployment Considerations*, (Macmillan Technology Publishing, Indianapolis, IN, ISBN 1-57870-166-X, 1999).

[Comer00] Comer, D. E., *Internetworking with TCP/IP, Principles, Protocols, and Architectures*, Fourth Edition, (Prentice Hall, Upper Saddle River, NJ 07458, ISBN: 0-13- 018380-6, 2000).

[ECCTLS] Gupta, V. *ECC Cipher Suites for TLS.* Draft Internet Engineering Task Force, Request for Comment, October 2004. http://www.ietf.org/proceedings/04aug/I-D/draft-ietf-tls-ecc-06.txt

[Fanto2002] Fanto, M, SSL Web Server and Client Configuration, PKI Technical Working Group Presentation. http://csrc.nist.gov/pki/twg/y2002/presentations/twg-02-16.pdf

[FIPS46-3] FIPS 46-3, *Data Encryption Standard (DES)*[27], http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

[FIPS81A] National Institute of Standards and Technology, *DES Modes of Operation,* Federal Information Processing Standard 81, 1980 December 2, http://csrc.nist.gov/publications/fips/fips81/fips81.htm

[FIPS81B] National Institute of Standards and Technology, *DES Modes of Operation Change Notice 2,* Federal Information Processing Standard 81, 1996 May31, http://csrc.nist.gov/publications/fips/fips81/fips81change2.pdf

[FIPS81C] National Institute of Standards and Technology, *DES Modes of Operation Change Notice 3,* Federal Information Processing Standard 81, http://csrc.nist.gov/publications/fips/fips81/fips81change3.pdf

[FIPS140-1] FIPS 140-1, *Security Requirements For Cryptographic Modules*, http://csrc.nist.gov/publications/fips/fips140-1/fips1401.pdf

[FIPS140-2] FIPS 140-2, *Security Requirements For Cryptographic Modules,* http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

[FIPS140Impl] National Institute of Standards and Technology, *Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program*, April 28, 2004, http://csrc.ncsl.nist.gov/cryptval/140-1/FIPS1402IG.pdf

[FIPS180-2] National Institute of Standards and Technology, *Secure Hash Standard (+ Change Notice to include SHA-224)*, Federal Information Processing Standards Publication 180-2, August 1 2002, http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf

[27] FIPS 46-3 is in the process of being withdrawn and replaced by NIST Special Publication 800-67, currently available at Hhttp://csrc.nist.gov/publications/nistpubs/800-67/SP800-67.pdfH

[FIPS186-2] National Institute of Standards and Technology, *Digital Signature Standard*[28], Federal Information Processing Standard 186-2, 27 January 2000, http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf.

[FIPS197] National Institute of Standards and Technology, *Advanced Encryption Standard (AES),* Federal Information Processing Standard 197, November 26, 2001 http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[FIPS198] National Institute of Standards and Technology, *The Keyed-Hash Message Authentication Code (HMAC)*, Federal Information Processing Standard 198, 6 March 2002, http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf.

[Hall00] Hall, E. A., *Internet Core Protocols, The Definitive Guide*, (O'Reilly & Associates, ISBN: 1-56592-572-6, February 2000).

[Hollenbeck04] Hollenbeck, S., *Transport Layer Security Protocol Compression Methods*, Internet Engineering Task Force, Internet Draft, 16 January 2004. Transport Layer Security Protocol Compression Methods, http://www.ietf.org/internet-drafts/draft-ietf-tls-compression-07.txt (expires July 16, 2004). (Note: See http://www.ietf.org/html.charters/tls-charter.html for updated drafts and status.)

[Housley01] Housley, R. and Polk, T., *Planning for PKI, Best Practices Guide for Deploying Public Key Infrastructure*, (John Wiley & Sons, New York, NY, ISBN 0-471-39702-4, 2001).

[KeyMgmt03] National Institute of Standards and Technology's Computer Security Resource Center, *Key Management Information,* http://csrc.ncsl.nist.gov/CryptoToolkit/tkkeymgmt.html.

[PKCS1] RSA Laboratories, *PKCS #1 v2.1: RSA Cryptography Standard*, 14 June 2002.

[Polk03] Polk, W., Hastings, N., and Malani, A., *Public Key Infrastructures that Satisfy Security Goals,* IEEE Internet Computing, Volume 7, Number 4, July-August, 2003.

[Rescorla01] Rescorla, E., *SSL and TLS – Designing and Building Secure Systems*, (Addison- Wesley, Upper Saddle River NJ, 07458, ISBN 0-201-61598, March 2001).

[RFC2246] Dierks, T. and Allen, C., *The TLS Protocol Version 1.0*, Internet Engineering Task Force, Request for Comment 2246, January 1999, http://www.ietf.org/rfc/rfc2246.txt

[RFC3268] Chown, P., *Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS),* Internet Engineering Task Force, Request for Comment 3268, June 2002, http://www.ietf.org/rfc/rfc3268.txt

[SP800-23] NIST Special Publication 800-23, *Guidelines to Federal Organizations on Security Assurance and Acquisition/Use of Tested/Evaluated Products,* August 2000, http://csrc.nist.gov/publications/nistpubs/800-23/sp800-23.pdf.

[28] FIPS 186-2 will be replaced by FIPS 186-3, which is currently out for comment.

[SP800-32] NIST Special Publication 800-32, *Introduction to Public Key Technology and the Federal PKI Infrastructure,* February 2001, http://csrc.nist.gov/publications/nistpubs/800-32/sp800-32.pdf

[SP800-36] NIST Special Publication 800-36, *Guide to Selecting Information Technology Security Products,* October 2003, http://csrc.nist.gov/publications/nistpubs/800-36/NIST-SP800-36.pdf.

[Stallings98] Stallings, W., *SSL: Foundation for Web Security*, Internet Protocol Journal, Volume1, Number 1, June 1998, http://www.cisco.com/en/US/about/ac123/ac147/archived_issues/ipj_1-1/ssl.htmlT

[X9.31] American National Standards Institute, *Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (Appendix A.2.4, Generating Pseudorandom Numbers) X9.31*, 1998.

[7498] ISO/IEC 7498-1: 1994(E), ITU-T Rec. X.200 (1994 E), Information Processing Systems - OSI Reference Model - The Basic Model.

NIST Special Publication 800-52

Guidelines for the Selection and Use of Transport Layer Security (TLS) Implementations

NIST
National Institute of Standards and Technology
Technology Administration
U.S. Department of Commerce

C. Michael Chernick, Charles Edington III, Matthew J. Fanto, Rob Rosenthal

COMPUTER SECURITY

June 2005

NIST Special Publication 800-52

Guidelines for the Selection and Use of Transport Layer Security (TLS) Implementations

Recommendations of the National Institute of Standards and Technology

C. Michael Chernick, Charles Edington III, Matthew J. Fanto, Rob Rosenthal

C O M P U T E R S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

June 2005

U.S. Department of Commerce
Donald L. Evans, Secretary

Technology Administration
Phillip J. Bond, Under Secretary of Commerce for Technology

National Institute of Standards and Technology
Arden L. Bement, Jr., Director

Table of Contents

Tables

Figures

Executive Summary

Office of Management and Budget (OMB) Circular A-130, *Management of Federal Information Resources,* requires managers of publicly accessible information repositories or dissemination systems that contain sensitive but unclassified data to ensure sensitive data is protected commensurate with the risk and magnitude of the harm that would result from the loss, misuse, or unauthorized access to or modification of such data. Given the nature of interconnected networks and the use of the Internet to share information, protection of this sensitive data can become difficult if proper mechanisms are not employed to protect the data. Transport layer security (TLS) provides such a mechanism to protect sensitive data during electronic dissemination across the Internet.

TLS is a protocol created to provide authentication, confidentiality and data integrity between two communicating applications. TLS is based on a precursor protocol called "The Secure Sockets Layer Version 3.0" (SSL 3.0) and is considered to be an improvement to SSL 3.0. SSL is specified in an expired Internet Draft working document of the Internet Engineering Task Force. Transport Layer Security Version 1 (TLS 1.0) specification is an Internet Request for Comments [RFC2246]. Each document specifies a similar protocol that provides security services over the Internet. While TLS 1.0 is based on SSL 3.0, and the differences are not dramatic; they are significant enough that TLS 1.0 and SSL 3.0 do not interoperate. This Special Publication provides guidance to the selection and implementation of the TLS protocol while making effective use of Federal Information Processing Standards (FIPS) approved cryptographic algorithms, and suggests that TLS 1.0 configured with FIPS based cipher suites is the appropriate secure transport protocol.[1] Use of the recommendations provided in this Special Publication would promote:

- More consistent use of authentication, confidentiality and integrity mechanisms for the protection of information transport across the Internet;
- Consistent use of recommended cipher suites that encompass FIPS algorithms and open source technology; and
- Informed decisions by system administrators and managers in the integration of transport layer security implementations.

While these Guidelines are primarily designed for Federal users and system administrators to adequately protect sensitive but unclassified Federal Government data against serious threats on the Internet, they may also be used within closed network environments to segregate data. (The client-server model and security services discussed also apply in these situations). This Special Publication does not supercede any existing NIST publication and should be used in conjunction with existing policies and procedures.

---

[1] While SSL 3.0 is the most secure of the SSL protocol versions, it is not approved for use in the protection of Federal information because it relies in part on the use of cryptographic algorithms that are not FIPS-Approved. TLS when properly configured is approved for the protection of Federal information.

Reports on Information Security Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive, unclassified information in Federal computer systems. This Special Publication 800-series reports on ITL's research, guidance, and outreach efforts in computer security and its collaborative activities with industry, government, and academic organizations.

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

National Institute of Standards and Technology Special Publication 800-52
Natl. Inst. Stand. Technol. Spec. Publ. 800-52, 33 pages (June 2005)
CODEN: NSPUE2

Acknowledgments

The authors, Michael Chernick and Matt Fanto of NIST, and Charles Edington and Robert Rosenthal of Booz Allen Hamilton (BAH) would like to thank the many people who assisted with the development of this document. In particular we would like to acknowledge Marcus Sills of BAH who aided with researching the complexities of the TLS protocol and Elaine Barker of NIST who gave guidance on random number generation.

1 Introduction

Today, many World Wide Web browsers and server applications rely on secure SSL and TLS communications to protect sensitive data transmitted through the Internet. Many books such as [Rescorla01], [Comer00], and [Hall00] describe the Internet's client-server model and communication protocol design principles. None guide Federal users and system administrators to adequately protect sensitive but unclassified Federal Government data against the most serious threats on the World Wide Web – eavesdropping, data tampering and message forgery. Other books such as [Adams99] and [Housley01] as well as technical journal articles (e.g., [Polk03]) and NIST publications (e.g., [SP800-32]) describe how Public Key Infrastructure (PKI) can be used to protect information in the Internet.

It is assumed that the reader of these Guidelines is somewhat familiar with the ISO seven-layer model communications model (also known as the seven-layer stack) [7498], as well as the Internet and public key infrastructure concepts, including, for example, X.509 certificates. If not, the reader may refer to the references cited above in the first paragraph of this introduction for further explanations of background concepts that cannot be fully explained in these Guidelines.

These Guidelines briefly introduce computer communications architectural concepts. The Guidelines place the responsibility for communication security at the Transport layer of the OSI seven-layer communications stack, not within the application itself. Protection of sensitive but unclassified Government information can adequately be accomplished at this layer when appropriate protocol options are selected and used by clients and servers relying on transport layer security.

Unfortunately, security is not a single property possessed by a single protocol. Rather, security includes a complex set of related properties that together provide the required information assurance characteristics and information protection services. Security requirements are usually derived from a risk assessment to the threats or attacks an adversary is likely to mount against a system. The adversary is likely to take advantage of implementation vulnerabilities found in many system components including computer operating systems, application software systems, and the computer networks that interconnect them. These guidelines focus only on security within the network, and they focus directly on the small portion of the network communications stack that is referred to as the transport layer.

Usually, the best defense against telecommunications attacks is to deploy security services implemented with mechanisms specified in standards that are thoroughly vetted in the public domain and rigorously tested by third party laboratories, by vendors, and by users of commercial off-the-shelf products.

Three services that most often address network user security requirements are confidentiality, message integrity and authentication. A confidentiality service provides assurance that data is kept secret, preventing eavesdropping. A message integrity service provides confirmation that data modification is always detected thus preventing undetected deletion, addition, or modification of data. An authentication service provides assurance of the sender or receiver's identity, thereby preventing forgery.

2 Security in a Layered Communications Architecture

The layering of computer communication protocols into a protocol stack (as defined by the OSI Seven Layer Model [7498]) enables developers to design new communication systems using already defined protocols and specific communication requirements within each layer of the stack. Each layer of the transmitting system communicates with the corresponding layer on the receiving system(s). Within this communications stack, the functionality of each layer is, in theory, independent of each other layer. Placement of security services and implementation of the security mechanisms within the stack is specific to each individual layer of the stack. Since the OSI Seven Layer Model does not explicitly define where security services are to be placed, there has been some debate over the exact placement of security services and other implementation mechanisms. These debates continue as new standards evolve to meet the communication needs of users, local and wide area networking vendors, Internet Service Providers (ISPs), and World Wide Web application designers.

Today we know that data privacy, integrity, and authenticated message delivery are required in a client-server model, where a user's client browser accesses one or more web server's applications. In this model, the inter-network "fabric" of telephone lines, network routers, firewalls, and other network components is seldom under the control of the end user's client software or of the server's application software; and so, data protection against eavesdropping, tampering or message forgery must be placed within the client/server protocols higher in the stack above this inter-network fabric. This ensures that security services are controlled jointly by the client and server, and not by the "fabric".

In a typical Internet architecture, the protocols in this fabric are the Transmission Control and Internet Protocol (TCP/IP) stack plus the protocols below IP in the stack. Protocols below IP include local area network (LAN) protocols or other link protocols such as dial up, or directly connected modems, fiber optic links, or satellite links. The TCP/IP stack provides for the transmission of packets through complex arrangements of local, wide, or metropolitan area or globally connected sets of inter- or intra-networking technology.

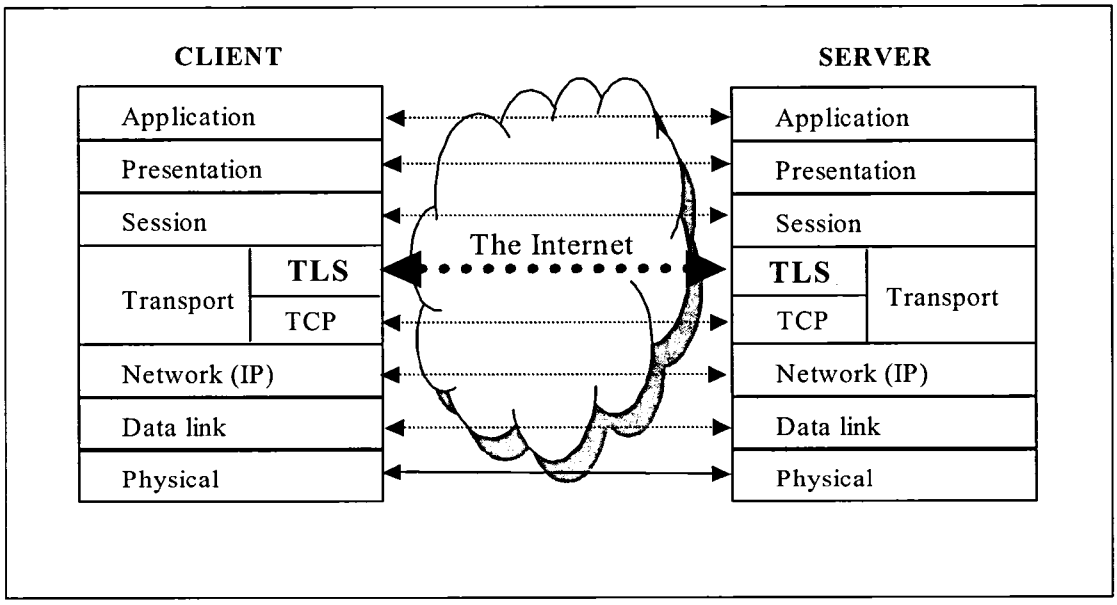

Figure 1: Client/Server Model[2]

Figure 1 depicts this architecture in the context of a client-server model. The arrow between the TLS protocol entities in Figure 1 emphasizes that the TLS entities make no assumptions about the security services provided by the interactions of the protocol entities lower in the stack. All of the security services needed to prevent eavesdropping, tampering or message forgery are provided by the TLS entities themselves. TLS does however rely on the communications functionality of the lower layer protocols to provide end-to-end reliable[3] delivery of data.

It is worth noting that TLS is not the only place in this architectural model where security services could be provisioned. As a general rule, placing services lower in the stack is advantageous because more higher layer entities can utilize the same functionality. However, placing security services lower in the model often requires sharing responsibility for security among many administrative organizations including the local, wide, or metropolitan area network and Internet Service Providers. Many feel that placing security in the transport layer, closer to, or part of, the client's browser and

[2] For simplicity and clarity firewalls and ISPs are not shown in this figure although in most deployed systems, firewalls and ISPs are used for connection to the Internet.

[3] Reliable delivery of data in this context has no security implication. Rather, reliable delivery of data implies that all messages presented to the sending TCP/IP stack are delivered in proper sequence by the receiving TCP/IP stack. These messages may be broken up into packets and fragmented or segmented as they are sent and routed through any arrangement of local, wide area or metropolitan networks. In general, as they are sent and routed through networks, the data are augmented with cyclical redundancy checks or forward error correction techniques to help ensure that the delivered messages are identical to the transmitted messages. Reliable delivery implies that the messages are properly reassembled and presented in correct order (proper sequence) to the peer protocol TLS entity.

server's application, is more appropriate for web-based applications rather than relying on or placing security services in lower levels.

2.1 Security in the Transport Layer

The Netscape Corporation[4] designed a protocol known as the Secure Sockets Layer (SSL) to meet security needs of client browsers and server applications. Version 1 of SSL was never released. Version 2 (SSL 2.0) was released in 1994 but had well-known security vulnerabilities. Version 3 (SSL 3.0) was released in 1995 to address these vulnerabilities. During this timeframe, Microsoft Corporation released a protocol known as Private Communications Technology (PCT), and later released a higher performance protocol known as the Secure Transport Layer Protocol (STLP). PCT and STLP never commanded the market share that SSL 2.0 and SSL 3.0 commanded. The Internet Engineering Task Force (IETF) (a technical working group responsible for developing Internet standards to ensure communications compatibility across different implementations), attempted to resolve, as best it could, security engineering and protocol incompatibility issues between the protocols. The IETF standards track Transport Layer Security Protocol Version 1.0 (TLS 1.0) emerged and was codified by the IETF as [RFC2246]. While TLS 1.0 is based on SSL 3.0, and the differences between them are not dramatic, they are significant enough that TLS 1.0 and SSL 3.0 do not interoperate. However, TLS 1.0 does incorporate a mechanism by which a TLS 1.0 implementation can negotiate to use SSL 3.0 with requesting entities as if TLS were never proposed. However, because SSL 3.0 is not approved for use in the protection of Federal information, (Section 7.1 of [FIPS140Impl]), TLS must be properly configured to ensure that the negotiation and use of SSL 3.0 never occurs when Federal information is to be protected.

These Guidelines attempt to make clear the impact of selecting and using secure web transport protocols for use in protecting sensitive but unclassified U.S. Government information.

2.2 The Security Parts of Transport Layer Security

Both the TLS 1.0 and SSL 3.0 protocol specifications use cryptographic mechanisms to implement the security services that establish and maintain a secure TCP/IP connection. The secure connection prevents eavesdropping, tampering, or message forgery. Implementing data confidentiality with cryptography (encryption) prevents eavesdropping; generating a message authentication code with a secure hash function prevents undetected tampering; and, authenticating clients and servers with public key cryptography-based digital signatures prevents message forgery. In each case – preventing eavesdropping, tampering and forgery – a key or shared secret is required by the cryptographic mechanism. A pseudo-random number generator and a key establishment algorithm provide for the generation and sharing of these secrets.

[4] Commercial company names are used for historical reference purposes only. No product endorsement is intended or implied

The rows in Table 1 identify the key establishment, confidentiality, digital signature and hash mechanisms currently in use today in TLS 1.0 and SSL 3.0. For the purpose of these Guidelines, Table 1 identifies which key establishment, confidentiality, and signature algorithms and which hash functions are Federal Information Processing Standards (FIPS). These FIPS form the basis of the recommendations in these Guidelines.

Table 1: Mapping The Security Parts of TLS to Federal Standards

Mechanism	SSL (3.0)	TLS 1.0	FIPS Reference
Key Establishment	RSA DH-RSA DH-DSS DHE-RSA DHE-DSS DH-Anon Fortezza-KEA	RSA DH-RSA DH-DSS DHE-RSA DHE-DSS DH-Anon	
Confidentiality	IDEA-CBC RC4-128 3DES-EDE-CBC Fortezza-CBC	IDEA-CBC RC4-128 3DES-EDE-CBC Kerberos AES	 FIPS 46-3, FIPS 81 FIPS 197
Signature	RSA DSA	RSA DSA EC*	FIPS 186-2 FIPS 186-2 FIPS 186-2
Hash	MD5 SHA-1	MD5 SHA-1	 FIPS 180-2, FIPS 198

Note: DES and all of the "export" algorithms of small key sizes (RC4-40, RC2-CBC-40, DES-40, DHE-DSS-Export and DHE-RSA-Export) have been left out of this table as these are now deprecated.

*An Internet-Draft proposing cipher suites containing Elliptic Curve (EC) algorithms has been introduced in the IETF. See [ECCTLS].

Both TLS 1.0 and SSL 3.0 package one key establishment, confidentiality, signature and hash algorithm into a "cipher suite." Not all combinations from the table work together. Instead, TLS 1.0 and SSL 3.0 implementations contain carefully crafted cipher suites that are registered by the IETF (in the case of TLS 1.0) and the Netscape Corporation (in the case of SSL).[5]

[5] The transport layer security specification assigns a 16-bit (4 hexadecimal digit) number to the defined cipher suite. For example: Ephemeral Diffie-Hellman (DHE) **key agreement**, Digital Signature Algorithm (DSA) **signature**, Triple Data Encryption Standard (3DES) using Encryption-Decryption-Encryption (EDE) and Cipher Block Chaining (CBC) **confidentiality** and the Secure Hash Algorithm (SHA-1) **hash** is assigned the hexadecimal value {0x00, 0x13}. (Note that this suite is not frequently used.) Of special note is the TLS 1.0 requirement that, "In the absence of an application profile standard specifying otherwise, a TLS compliant application MUST implement the cipher suite TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA.", which is represented by this example. However, the current draft of TLS 1.1 mandates TLS_RSA_WITH_3DES_EDE_CBC_SHA, and this suite is more commonly used.

A given implementation of the standard TLS 1.0 or SSL 3.0 protocol may implement one or more cipher suites from the registered set of suites. Finding a common cipher suite between a client and a server is accomplished through a built-in protocol "handshake negotiation" mechanism described in section 3, Negotiating Security Options, on page 13.

2.2.1 Key Establishment[6]

The following key establishment algorithms are used in various cipher suites.

RSA – The sender generates a random session key (pre-master secret) and encrypts it under the recipient's public key. The session key is a symmetric key.

DH (Diffie-Hellman) – The sender and receiver each have key pairs. To compute an agreed-upon session key, the sender combines his private key with the receiver's public key. The receiver combines his private key with the sender's public key.

There are three different types of DH communications, Static (also known as Fixed), Ephemeral, and Anonymous. These are described as follows[7]:

- **Fixed Diffie-Hellman:** This is a Diffie-Hellman key exchange in which the server's certificate[8] contains the Diffie-Hellman public parameters signed by the certificate authority (CA). That is, the public-key certificate contains the Diffie-Hellman public-key parameters. The client provides its Diffie-Hellman public key parameters either in a certificate, if client authentication is required, or in a key exchange message. This method results in a fixed secret key between two peers, based on the Diffie-Hellman calculation using the fixed public keys.

- **Ephemeral Diffie-Hellman:** This technique is used to create ephemeral (temporary, one-time) secret keys. In this case, the Diffie-Hellman public keys are exchanged, and signed using the sender's private RSA or DSS key. The receiver can use the corresponding public key to verify the signature. Certificates are used to authenticate the public keys. This option appears to be the most secure of the three Diffie-Hellman options because it results in a temporary, authenticated key.

---

[6] "Key Establishment" is the process of establishing a shared secret key used for encrypting data exchanged between client and server over a TLS connection. Key establishment is often called "key exchange". In some key establishment schemes (e.g., RSA), the client generates a random key and sends it to the server. In other schemes (e.g., Diffie-Hellman) the server generates some random data, sends the data to the client, the client generates additional random data, combines in with the server's random data, and the resulting key is sent to the server to be used as a secret key. This latter scheme is an example of a "key agreement" type of key establishment because the two sides together agree on a key.

[7] The Diffie-Hellman descriptions are used with the permission of Dr. William Stallings, and appeared in [Stallings98]

[8] For brevity, the term "certificate" is used in these guidelines to mean "X.509 certificate".

- **Anonymous Diffie-Hellman:** The base Diffie-Hellman algorithm is used, with no authentication. That is, each side sends its public Diffie-Hellman parameters to the other, with no authentication. This approach is vulnerable to man-in-the-middle attacks, in which the attacker conducts anonymous Diffie-Hellman exchanges with both parties.

Fortezza-KEA – KEA was the key agreement algorithm used by the Fortezza card supported by the Department of Defense, and KEA was originally classified. It exists in SSL 3.0, but the IETF standards committee did not include it in TLS 1.0.

The rules and protocols for generating and establishing keys, and the handling of those keys throughout their lifecycle, directly affects the level of security achieved. The security and reliability achieved depends on the strength of the key generation process and the protection afforded to those keys. Secret keys and the private key of a public key pair must be protected from disclosure, modification, substitution, and unauthorized deletion.

Some aspects of the key life cycle are addressed by the selection of appropriate cipher suites (i.e., the key establishment algorithm itself) or by the selection of validated modules[9] (e.g., primitives for key generation, storage, and destruction).

In deploying certificates for the support and use of TLS 1.0 and SSL 3.0 it is important to recognize that associated keys must be protected. Users must plan for and deal with many of these generally accepted cryptographic key life-cycle issues:

1) **Generation** of keys is accomplished with the aid of a pseudo-random number generator (PRNG). The protocol handshake sequence "ClientKeyExchange" creates a shared secret of random bits called the pre-master secret.

2) **Establishment** of keys occurs after the "ClientKeyExchange" message used in the handshake sequence creates the shared pre-master secret. This shared secret is expanded using a key derivation function in the client and the server. The key derivation function establishes the individual cryptographic keys that will be used for the various encryptions, authentication and the secure hash functions.[10]

3) **Storage** of keys and the shared secrets used to generate the keys is an issue in transport layer security implementations. Usually, the implementation relies on the capabilities of the operating system to protect this sensitive storage area.

4) **Crypto-periods** of both the session keys and certificate keys are not a transport layer security issue. The established session keys have a lifetime as long as the

[9] Cryptographic modules for Federal Government use must be validated in accordance with [FIPS140-1] or [FIPS140-2].

[10] The key derivation function uses the pre-master key, a label and a seed composed of shared random numbers to develop a master secret from which enough keying material is generated for all the algorithms defined in the cipher suite. This might include up to six keys – an encryption key, a MAC key (see section 2.2.5 for a description of Message Authentication Codes (MACs)) and an initialization vector for both the client and server.

session lasts.[11] Any keys used in the optional X.509 certificates that identify clients and servers have a crypto period that is determined by the policy of the certificate issuing authority.

5) **Recovery** of session keys is not a transport layer security issue. There is no requirement for a key recovery mechanism in the transport layer security protocol because the shared session key is ephemeral, a new session can be easily established with a new ephemeral key.[12]

6) **Destruction** of all of the state variables including the session keys occurs when the session ends. The protocol implementation relies on the operating system to insure that there is no reuse of storage areas, which may contain sensitive information.

2.2.2 Confidentiality

The following symmetric encryption algorithms are used in various cipher suites to provide confidentiality:

IDEA[13] – IDEA is a block cipher that operates on 64 bit plaintext blocks. The key is 128 bits long. The same algorithm is used for encryption and decryption. IDEA is not FIPS-approved.

RC4 – RC4 is a stream cipher that uses a variable length key of anywhere between 8 and 2048 bits long. RC4 is not FIPS-approved.

3DES-EDE – The Data Encryption Standard (DES) is the most widely used symmetric block cipher. It uses 64 bit blocks and a 56-bit key. Triple DES (also known as 3DES) super-encrypts by running the data through the DES algorithm 3 times with different keys. The first time it **E**ncrypts with key 1, the second time it **D**ecrypts with key 2 and the third time it **E**ncrypts again with key 3; hence the acronym 3DES-EDE. 3DES-EDE is FIPS-approved.

AES – The Advanced Encryption Standard is a FIPS-approved symmetric block cipher encryption algorithm that may be used by U.S. Government organizations (and others) to protect sensitive but unclassified information. AES uses 128, 192, or 256 bit keys, however cipher suites have only been defined for 128 and 256 bit keys to reduce the over proliferation of cipher suites. The block

---

[11] The message sequence space for SSL 3.0 and TLS 1.0 is 64 bits long; so, there is a theoretical limit of 2^{64} messages per session. If this limit is reached the connection must close as the sequence number cannot repeat or recycle. Furthermore, there is no mechanism to "rekey" the session. In effect this limits the crypto period of the session key.

[12] Servers and clients may (and often do) cache the master secret (but not the session key) to reduce the significant overhead in session resumption. After some reasonable timeout period, the master secret should be destroyed on both the server and the client.

[13] Cipher Block Chaining (CBC) is one of four DES modes of operation and can be used with all of the symmetric block ciphers listed here (i.e., IDEA, 3DES-EDE, and AES). CBC chains together the ciphertext of one block with the plaintext of the next block. Additional information on CBC can be found in [FIPS 46-3].

size in AES is 128 bits. The AES algorithm [FIPS197] is designed to replace DES and 3DES. AES is FIPS-approved.

Note that RC4 is currently the most commonly used confidentiality algorithm in SSL/TLS. However, RC4 is not FIPS-approved.

2.2.3 Signature

The following digital signature algorithms are used in various cipher suites:

RSA – To sign a message with the RSA algorithm the signatory computes a message digest or hash of the message and encrypts it with the private key. To verify an RSA signed message, the verifier decrypts the message digest with the signatory's public key and compares it with a locally computed hash of the original message. If the decrypted hash matches the locally calculated hash, the signature is valid. The use of RSA for digital signatures requires encryption and decryption.

DSA – To sign a message with the DSA algorithm, the signatory computes a signature using the SHA-1 hash algorithm and the signatory's private key. To verify a DSA signature, the verifier performs a computation using the message hash, the signatory's signature and the public key. The DSA verifier returns a yes or no rather than a decrypted hash as in the RSA signature. There is no encryption or decryption performed.

2.2.4 Hash

Hash algorithms used for cryptography (one-way hashes) have the property that it is computationally infeasible to find two messages that hash to the same value. When a message of any length is input to an algorithm, the result is an output called a message digest. The message digests range in length, depending on the algorithm used, however for use within Government environments four lengths have been specified; 160 bits, 256 bits, 384 bits, and 512 bits. Hash algorithms are identified as being secure if for a given algorithm, it is computationally infeasible that a message corresponding to a given message digest will be found, as well as finding two different messages producing the same message digest. Any change to a message will, with a very high probability, result in a very different message digest. This will cause a verification failure when the hash algorithm is used in conjunction with a digital signature algorithm or a keyed-hash message authentication algorithm.

The following hash algorithms are used in various cipher suites:

SHA–1: An algorithm for computing a condensed representation of a message or a data file. When a message of any length less than 2^{64} bits is input, SHA-1 produces a 160-bit output called a message digest. (The 2^{64} bit limit is caused by a 64-bit field size for a length descriptor used by SHA-1.)

MD5: When a message of any length is input, MD5 produces a 128-bit message digest. MD5 is not a FIPS-approved hash function, but it is in common use.

Note: Currently SHA-256, SHA-384, and SHA-512 are not featured in standard SSL or TLS cipher suites. It is expected that these variants of the secure hash algorithm will be added to TLS.

2.2.5 MAC

Message authentication (and integrity checking) is often achieved through the construction of a message authentication code (MAC), which is a small amount of additional data that is sent along with a message. To use a MAC, the sender and recipient must have a shared secret key known to no one else. Before sending a message, the sender computes the MAC as a function of the message and the key. Then the sender sends the message along with the computed MAC. The recipient computes a new MAC for the received message using the message itself, the secret key, and the same function as the sender. If the newly computed MAC is the same as the MAC received with the message, the recipient can be sure that the message has not been modified in transit, and that the sender knows the same secret key. That knowledge of the secret key assures the recipient of the sender's identity.

Cryptographic hash functions (with properties as described in 2.2.4 above) are often used as the function used to create MACs. In a simple scheme the secret key is simply prepended to the message, and the hash of this concatenation is used as the MAC. Message authentication codes generated using hash functions are known as HMACs and are used in TLS, SSL, and other common security mechanisms. For a detailed description of HMAC operation see [RFC2104] and [FIPS198].

A number of operations in the TLS protocol require an HMAC for authentication and integrity. Forging these MACs is infeasible without knowledge of the MAC secret key thus providing the necessary level of trust between communicating parties. Although HMACs can be used with a variety of different hash algorithms, TLS only specifies use of MD5 and SHA-1. To provide additional security for handshakes within the protocol, TLS uses both MD5 and SHA-1 in conjunction with each other to produce the HMAC.

3 Negotiating Security Options

Figure 2 provides details of the transport layer security protocol entities. The entities have been developed to allow control information to flow between the client and the server. Three types of control messages are exchanged: "Handshake", "ChangeCipherSpec, and "Alert". In addition, the entities exchange application data between the client and server protected by the security services provisioned by the negotiated cipher suite. These security services are negotiated and established with the Handshake.

The "Handshake" consists of a series of message exchanges between the client and the server. This series of messages is used to establish the TLS/SSL environment, including a set of negotiated security algorithms. The ChangeCipherSpec message is used to inform the other side to begin using the negotiated security services by changing to the cipher suite agreed to during the Handshake. All messages sent after ChangeCipherSpec message are encrypted using the just negotiated cipher suite.

The Alert provides a way to signal special security channel events and errors, as well as asynchronous events that may occur during a TLS/SSL session. In addition, an alert message is used to terminate a session.

Specific details of the Handshake, ChangeCipherSpec and Alert message exchanges are beyond the scope of these Guidelines; but understanding that the Handshake negotiates a cipher suite is important to understanding how to configure and use Transport Layer Security.

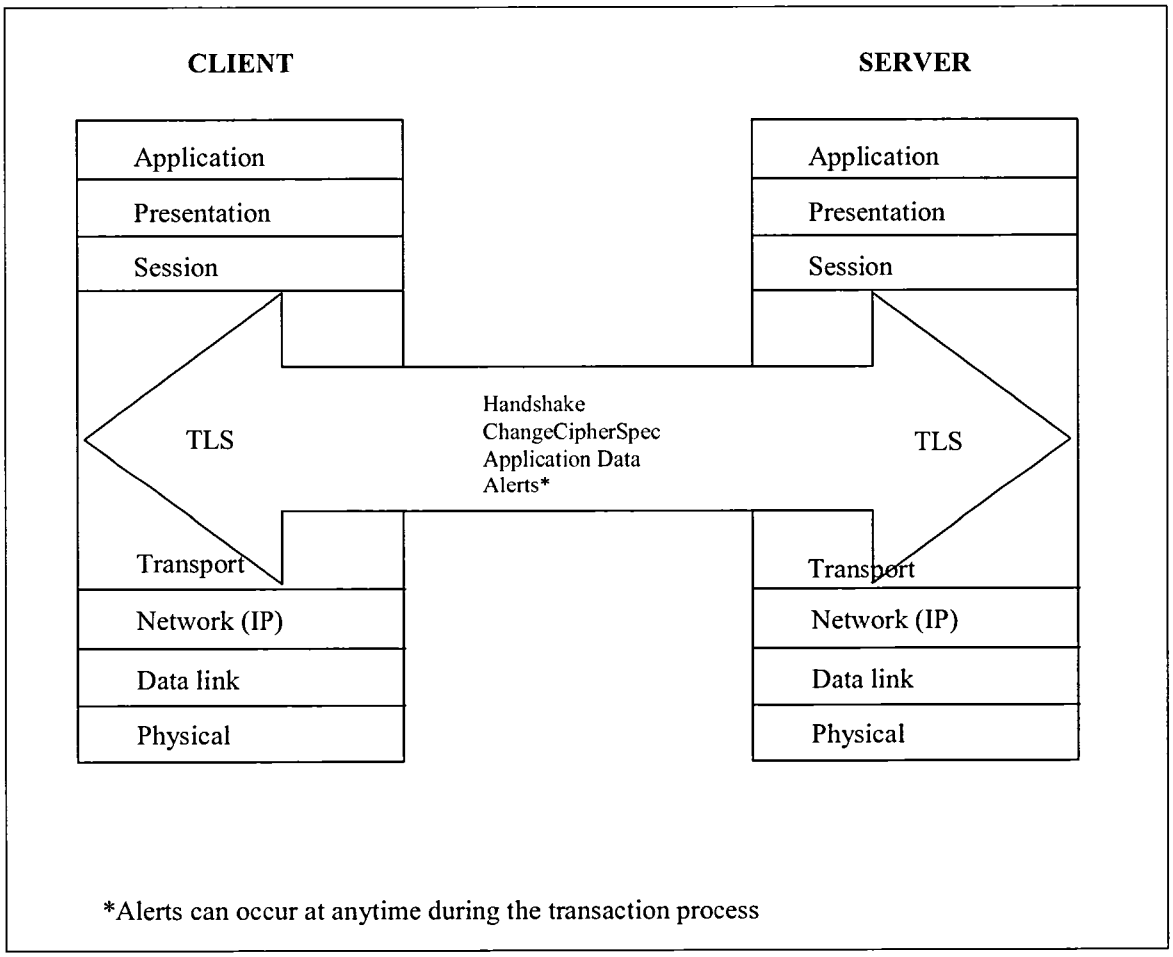

Figure 2: The Transport Layer Security Protocol Entity

The Transport Layer Security handshake protocol establishes a secure channel inside of a TCP/IP connection before passing any data from the application.

- The handshake protocol initializes both the client and server to use optional cryptographic capabilities by negotiating a cipher suite of algorithms and functions including key establishment, digital signature, confidentiality and integrity algorithms with their respective key sizes, and hash functions. This negotiation begins with the "ClientHello" message and continues with the "ServerHello" message.
- The handshake protocol may exchange public key digitally signed X.509 certificates[14] to optionally authenticate the server to the client and vice versa. In most cases, the server presents a certificate to the client, but the client does not present a certificate to the server. However, TLS and SSL allow for certificates to be presented by a server, by a client, by both, or by neither in negotiating a

[14] The use of X.509 certificates is fundamental to TLS/SSL, as well as other PKI-enabled services. For a comprehensive explanation of X.509 certificates see, for example, [Adams99] or [Housley01].

session. The important point to note is that presenting a valid X.509 certificate and proving possession of the private key authenticates the presenter to the recipient.

- Using the negotiated key establishment algorithm, the handshake protocol exchanges random data for developing keying material to be used by the cryptographic algorithms.

When all the security parameters are in place (i.e., after the Handshake), the "ChangeCipherSpec" message engages the negotiated cryptographic capabilities for application data exchanges.[15]

After the handshake protocol completes, application data may be exchanged between the client and server. This data is encrypted and integrity checked using the security services that were negotiated during the handshake.

Alerts are used to signal asynchronous/exceptional events during a TLS/SSL session. For example, an alert can be used to signal "decrypt_error" or "access_denied". Some alerts are used for warning, and others are considered "fatal" and can lead to immediate termination of the session. A "close_notify" alert is used to signal normal termination of a session. Like all other messages, after the handshake protocol is completed, alert messages are encrypted and optionally compressed.

3.1 The Cipher Suite

The negotiated algorithm identifiers are referred to collectively as the "Cipher Suite". Cipher suites are identified in human readable form using a mnemonic code. Each cipher suite includes a code for the protocol – either SSL or TLS. The next mnemonic identifies the key establishment algorithm and digital signature algorithm followed by the word "WITH" followed by the confidentiality algorithm followed by the hash function. An example is:

TLS_DHE_DSS_WITH_3DES-EDE-CBC_SHA

This cipher suite is for:

- A Transport Layer Security Version 1.0 protocol, TLS,
- The Ephemeral Diffie-Hellman key agreement, DHE,
- The Digital Signature Standard, DSS (which implies the Digital Signature Algorithm, DSA),
- The Triple Data Encryption Standard's Encrypt-Decrypt-Encrypt option in Cipher Block Chaining mode, 3DES-EDE-CBC, and
- The Secure Hash Algorithm, SHA-1 (used to compute a HMAC).

[15] Although not specifically related to security, the handshake also optionally negotiates a compression algorithm for the application data exchanges.

Both TLS (1.0) and SSL (3.0) specify registered index numbers for the cipher suites. When negotiating a cipher suite, the client (which always initiates TLS/SSL sessions) sends a handshake message with a list of cipher suite indexes it will accept. The server chooses from the list and sends a handshake message back indicating which cipher suite it will accept. Although the client orders the list with the "strongest" cipher suites listed first, the server may choose **ANY** of the cipher suites proposed by the client. Therefore there is **NO** guarantee that the negotiation will settle on the strongest suite in common. If no cipher suites are in common the connection is aborted.

When the negotiated options are complete, the client sends a "ChangeCipherSpec" and a "Finished" message. In response, the server also sends a "ChangeCipherSpec" and a "Finished" message to place the communication channel in a secure mode to protect transmission of the client's browser and server's application data.

3.2 Data Integrity of the Handshake

Data integrity is maintained throughout the handshake process and finally completed with the sending of the "Finished" message. A "Finished" message is always sent immediately after the "ChangeCipherSpec" message to verify the key exchange and authentication processes were successful. With successful exchanges of this message the client and server verify that the entire handshake has not been modified. This verification is possible because each side sends a hash of the concatenated handshake messages to the other side, which compares it to the same result computed locally. If the hash values differ, the handshake has been modified and the connection is aborted. If the hash values are the same, there is high assurance that the entire handshake has cryptographic integrity – nothing was modified, added or deleted.

4 Recommendations

This section presents criteria for developing specific recommendations when selecting, installing and using a transport layer security.

4.1 Selection Criteria

When implementing transport layer security mechanisms (usually web servers and browsers), there are several important aspects to consider such as whether the implementation:

- **Is standards-based** – The interaction between components in a transport layer security mechanism should be a well-defined communication protocol with no deviations. Additionally, FIPS-approved algorithms for authentication, encryption, and the generation of message digests should be used in all implementations.
- **Supports interoperability** – Any implementation should promote interoperability among components. The selection of a particular server solution should not prevent the use of any standards-based client or vise versa.
- **Includes evaluated products** – Key components of the implementation should be independently evaluated against known standards (e.g., cryptographic modules validated against FIPS 140-1 or 140-2).
- **Select important features** – The implementation should include those features that users consider most important to their operating environment.
- **Is Open Source** – The implementation should be an open source solution to help prevent being locked into a proprietary implementation that may not support interoperability or identified standards in the future.

4.2 Protocol Selection

Due to the vulnerabilities inherent in SSL 2.0, the only reasonable protocols to consider for deploying transport layer security are SSL 3.0 or TLS 1.0. However, using the criteria discussed in 4.1, TLS 1.0 is the only acceptable alternative. It is an IETF "standard" that incorporates FIPS-approved algorithms while SSL 3.0 is not standards-based, resulting in several competing incompatible SSL variants. Because SSL 3.0 uses some non-FIPS-approved cryptographic algorithms, it is not approved for use in the protection of Federal information [FIPS140Impl]. Likewise there exist incompatibilities between SSL 3.0 and TLS 1.0 implementations that prevent the integrated use of both TLS 1.0 and SSL 3.0 components, such as:

- SSL 3.0 allows a party to start sending application data as soon as that party's "Finished" message is sent. TLS requires a "wait" until the other "Finished" message is received.

- SSL 3.0 "ClientKeyExchange" contains two bytes representing the SSL version number. Some implementations use this value as a negotiation. Use of these two bytes as a negotiable value is incompatible with TLS 1.0.
- The key derivation from the pre-master secret is different. (SSL uses a key derivation model in which half the master secret is generated using only MD5, which is not a FIPS-approved hash algorithm.).
- SSL 3.0 omits the length bytes when RSA is used for sending the encrypted pre-master secret.
- Alert messages produce different results.

Although the selection of TLS may cause legacy problems with clients that are incapable of supporting TLS, almost all currently used clients (e.g., Internet Explorer, Netscape, Mozilla) support TLS.

4.3 Cipher Suite Selection

The last column in **Table 1: "Mapping the Security Parts of TLS to Federal Standards"** on page 7, lists the Federal Information Processing Standards that are applicable for use in various components of the negotiated cipher suites. Table 2 and Table 3 in Section 5 provide recommendations for the cipher suites that should be offered based, on the information presented below.

It is important to note that RFC 2246 specifies all TLS compliant applications implement the TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA cipher suite. This document does not supplant this requirement, but instead goes beyond RFC2246 and specifies only the following cipher suites may be used.

Key Establishment: For maximum security, RSA or DSA authentication with ephemeral Diffie-Hellman key agreement is recommended (e.g., AES might be TLS_DHE_RSA_WITH_AES-128_CBC_SHA and TLS_DHE_DSS_WITH_AES-256_CBC_SHA.).

These cipher suites provide perfect forward secrecy[16], and can support large keys for long-term confidentiality.

The Federal Government has not finalized a specification for FIPS-approved or NIST-Recommended algorithms for establishment of cryptographic keys. Note, however that a NIST Recommendation is currently under development. See [KeyMgmt03]. Upon completion, only recommended algorithms shall be used.

Confidentiality: Since TLS 1.0 is the security protocol of choice within the transport layer; only 3DES-EDE-CBC or AES should be offered during the handshake. Clients should never offer the deprecated 40 or 56-bit suites. While encryption is optional,

[16] Perfect forward secrecy is the condition in which the compromise of a session key or long-term private key after a given session does not cause the compromise of any earlier session key.

anonymous cipher suites are not allowed. For use in government agency to government agency communication, only FIPS-approved algorithms shall be used.[17]

Signature: The use of 1024 bit (key size) RSA/DSA server X.509 certificates is acceptable until the year 2010. If the client identity must be authenticated after the year 2010, key sizes larger than 1024 bits are needed.

Note that although RSA keys should have a key size of at least 1024 bits, many browsers and SSL/TLS servers have no mechanism to allow users to explicitly specify a minimum acceptable key size. However, by setting the symmetric key size to 128 bits (allowed by many browsers and servers) then an RSA key size of 1024 is often implicit in these browsers.

Hash: TLS offers two options for a cryptographic hash algorithm. These are SHA-1 and MD5. While SHA-1 is a FIPS-approved cryptographic hash algorithm, MD5 is not and thus should not be offered as a selection.

Note: The choice of a hash function within a cipher suite selection affects only the HMAC used to protect payload traffic. Other uses of hashes in key derivation or signatures are directly affected by cipher suite choice: RSA client authentication signature always uses both SHA-1 and MD5 while DSA client signatures only uses SHA-1.

[17] RC4 is acceptable for use on Government "Client" systems in very limited circumstances where secure information is to be transferred between Government systems and non-government servers, and 3DES or better (e.g., AES) is not supported by the server. For example, many vendor web sites providing supplies to the government support nothing stronger than RC4, and credit card information must be conveyed and secured to order supplies. In such cases risk is limited to exposure of government credit card information, and agencies may wish to take this risk to expedite ordering of supplies. RC4 should never be used on Government "Server" systems where government owned/generated data is to be made available in a secure manner to "client" systems.

5 Guidance

This section offers guidance to system administrators responsible for procuring, installing and maintaining implementations of transport layer security protocols.

Implementation Selection/Procurement – Among the responsibilities of any implementer or vendor of security functionality are:

- To provide quality random numbers for key generation,
- To protect the keying material and its storage,
- To properly implement and test key establishment, encryption, and signature algorithms and hash functions.

The procurement authority should ensure that the implementation meets a minimum set of universally accepted tests. Guidance for buying security products is provided in [SP800-23], Guidelines to Federal Organizations on Security Assurance and Acquisition/Use of Tested/Evaluated Products and [SP800-36], Guide to Selecting Information Technology Security Products.

Installation – Installation guidance usually involves following the vendor's general guidelines for configuration of the TLS or SSL protocol and following local (either to the server or to the client) configuration policy for selecting the optional protocol security services. For example, a client's local policy might state that server authentication is required. The system administrator would follow the vendor's prescribed methods for enabling client/server authentication.

The following security services should be configured and provided by the TLS implementation:

- Confidentiality,
- Data integrity,
- Peer entity authentication for clients and servers.

Appropriate cipher suites must also be selected and their priority ordered from strongest to weakest acceptable.

When peer entity authentication is selected, special attention should be paid to the public key infrastructure constructs used by TLS and SSL. Here, X.509 certificates are used. For example, for a client to be authenticated to a server, the authentication process requires that there must be a valid certification path that starts with one of the client's trust roots[18]. Part of the installation process is to ensure that all required trusted roots are present in the client and server implementations.

---

[18] A "trust root" (also known as a "trust anchor") is a X.509 certificate issued by (and signed by) a trusted authority. Before accepting a X.509 certificate (e.g., one presented by a TLS server) as valid, the user (known as a "relying party") must check to see that the certificate is signed by a trust root, or signed by an intermediate trusted authority whose certificate is signed by a trust root. There may be multiple intermediate trusted authorities, but the user must be able to find a chain of certificates that can traced back to a trust root before the user can rely upon a X.509 certificate. For a comprehensive explanation of X.509 certificates and trust see, for example, [Adams99] or [Housley01].

Note that for most common usage TLS and SSL are used to authenticate servers and not to authenticate clients. For merchandising implementations, for example, clients need to validate that they are dealing with an authenticated merchant before entering credit card information. The merchant only cares that a valid credit card is presented. (He does this by communicating with the credit card issuer. The details of that validation are beyond the scope of these guidelines.) The merchant does not need to authenticate the buyer.

Although outside of the installation process for the TLS implementation, another consideration is the adequate storage and protection of the client's and/or server's private key within a secure cryptographic module or token. This will prevent the masquerading of one party within the connection.

Maintenance – Once the server or client implementation is installed and operational, maintenance of the product generally follows local policies and operating procedures. For example, the site system administrator may be required to check for product updates and patches and install as needed. Within the local operating procedures, provisions need to be made for checking for and obtaining updated certificate revocation lists (CRLs) or using any on-line validation mechanisms available from the Certification Authority[19].

5.1 Considerations for Selecting TLS Client Implementations

Clients play a limited, but crucial role in the overall security posture. The client negotiates three parameters: the protocol version, the cipher suite, and the compression algorithm. These items are presented in the "ClientHello" message and form the basis for the server to negotiate the strongest possible security options.

The ClientHello message is the first message to be sent as the client establishes a TLS connection to the server. These messages allow the client to stay connected, re-establish an existing session, or to establish several independent secure sessions without repeating the full handshake procedure.

The client version field within the ClientHello message represents the protocol version that client supports. This field should contain the highest version number the client is prepared to support. For implementations that support TLS this value is: major=3, minor=1 (which represents 3.1, and hence TLS). All non-TLS implementations should use major=3, minor=0 for SSLv3. This designation does not limit the implementation to the identified protocol version. For example, if a client wishes to use only TLS, the client must connect to the server and is responsible for terminating the connection if the server selects any other protocol. Under no circumstances should a client use any protocol less than SSLv3[20]. For the most secure protection of data, only use clients that support TLS and that can disable all versions of SSL.

---

[19] Certificate authorities are trusted authorities that issue X.509 certificates to end users and to intermediate certificate authorities. When a certificate authority determines that a certificate has been compromised or should be considered invalid for further use, it adds the serial number of that certificate to its list of revoked certificates (i.e., to its CRL). Certificate authorities periodically distribute updated CRLs. Users (relying parties) should check the CRL before using an otherwise valid X.509 certificate to ensure that it has not been revoked.

[20] Only TLS can be used for the protection of Federal data. There may be instances where Federal users need access to non-Federal sites that do not support TLS. In these instances, Federal managers may allow the use of SSLv3 to transfer non-Federal data. However, the SSLv3 should be used in only in limited, low risk situations, and non-Federal sites should be encouraged to support TLS.

To ensure the security of the connection, client implementations should support the cipher suites presented in Table 2. This table presents the cipher suites listed in order of descending security strength. Unfortunately the order of these cipher suites is ignored by the server, which will select any cipher suite that it prefers from those offered by the client.

Table 2: Recommended Client Cipher Suites[21]

Cipher Suite	Authent-ication	Key Establishment	Encryption	Digest
TLS_DHE_DSS_WITH_AES_256_CBC_SHA	DSS	DHE	AES_256_CBC	SHA-1
TLS_DHE_RSA_WITH_AES_256_CBC_SHA	RSA	DHE	AES_256_CBC	SHA-1
TLS_RSA_WITH_AES_256_CBC_SHA	RSA	RSA	AES_256_CBC	SHA-1
TLS_DH_DSS_WITH_AES_256_CBC_SHA	DSS	DH	AES_256_CBC	SHA-1
TLS_DH_RSA_WITH_AES_256_CBC_SHA	RSA	DH	AES_256_CBC	SHA-1
TLS_DHE_DSS_WITH_AES_128_CBC_SHA	DSS	DHE	AES_128_CBC	SHA-1
TLS_DHE_RSA_WITH_AES_128_CBC_SHA	RSA	DHE	AES_128_CBC	SHA-1
TLS_RSA_WITH_AES_128_CBC_SHA	RSA	RSA	AES_128_CBC	SHA-1
TLS_DH_DSS_WITH_AES_128_CBC_SHA	DSS	DH	AES_128_CBC	SHA-1
TLS_DH_RSA_WITH_AES_128_CBC_SHA	RSA	DH	AES_128_CBC	SHA-1
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DSS	DHE	3DES_EDE_CBC	SHA-1
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA	RSA	DHE	3DES_EDE_CBC	SHA-1
TLS_RSA_WITH_3DES_EDE_CBC_SHA	RSA	RSA	3DES_EDE_CBC	SHA-1
TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA	DSS	DH	3DES_EDE_CBC	SHA-1
TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA	RSA	DH	3DES_EDE_CBC	SHA-1
TLS_RSA_WITH_RC4_128_SHA[22]	RSA	RSA	RC4_128	SHA-1

At this time, compression options have not been defined for either TLS[23] or any version of SSL. However, OpenSSL and some proprietary implementations support private compression algorithms. Care should be given to ensure that these proprietary and/or private algorithms, if implemented, do not weaken the security posture of the protocol. Also, many implementations do not support compression, so compression may not be possible during a TLS/SSL session, even if desirable.

5.2 Server Considerations

Cipher Suites

Although the client may present the cipher suites that it prefers in order of descending preference, the server generally does not defer to the client's preferred cipher suite. The server may, at its choosing, select a common cipher suite that it prefers. The following

---

[21] TLS has standardized AES cipher suites, however, it is expected that these cipher suites are not yet widely supported by commercial products. As products begin to provide support for AES, client implementations should support AES cipher suites as the highest priority cipher suites.

[22] RC4 is not a FIPS-approved cryptographic algorithm. For this reason, cipher suites with RC4 should be offered only when communicating with non-government entities in limited, low risk situations for the transfer of non-Federal data when a FIPS-approved encryption algorithm is not supported. Normally this cipher suite should not be offered.

[23] An Internet Draft has been proposed to define compression methods for TLS [Hollenbeck04].

table (Table 3) represents the cipher suites that a TLS server implementation should support. This table presents the cipher suites in order of descending preference.

Table 3: Recommended Server Cipher Suites[24]

Cipher Suite	Auth	Key Establishment	Encryption	Digest
TLS_DHE_DSS_WITH_AES_256_CBC_SHA	DSS	DHE	AES_256_CBC	SHA-1
TLS_DHE_RSA_WITH_AES_256_CBC_SHA	RSA	DHE	AES_256_CBC	SHA-1
TLS_RSA_WITH_AES_256_CBC_SHA	RSA	RSA	AES_256_CBC	SHA-1
TLS_DH_DSS_WITH_AES_256_CBC_SHA	DSS	DH	AES_256_CBC	SHA-1
TLS_DH_RSA_WITH_AES_256_CBC_SHA	RSA	DH	AES_256_CBC	SHA-1
TLS_DHE_DSS_WITH_AES_128_CBC_SHA	DSS	DHE	AES_128_CBC	SHA-1
TLS_DHE_RSA_WITH_AES_128_CBC_SHA	RSA	DHE	AES_128_CBC	SHA-1
TLS_RSA_WITH_AES_128_CBC_SHA	RSA	RSA	AES_128_CBC	SHA-1
TLS_DH_DSS_WITH_AES_128_CBC_SHA	DSS	DH	AES_128_CBC	SHA-1
TLS_DH_RSA_WITH_AES_128_CBC_SHA	RSA	DH	AES_128_CBC	SHA-1
TLS_DHE_DSS_WITH_AES_256_CBC_SHA	DSS	DHE	AES_256_CBC	SHA-1
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DSS	DHE	3DES_EDE_CBC	SHA-1
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA	RSA	DHE	3DES_EDE_CBC	SHA-1
TLS_RSA_WITH_3DES_EDE_CBC_SHA	RSA	RSA	3DES_EDE_CBC	SHA-1
TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA	DSS	DH	3DES_EDE_CBC	SHA-1
TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA	RSA	DH	3DES_EDE_CBC	SHA-1

Note that all server certificates with RSA keys should have a key length of at least 1024 bits.

Client Authentication

Client authentication when required is accomplished during the handshake. The server initiates client authentication by requesting the client's certificate and providing guidance as to the types of certificates and algorithms the server will accept. The exchange is completed when the client responds with its certificate and a signed hash of the original handshake to prove possession of the corresponding private key.

Although the protocol allows the server to continue the connection using another authentication mechanism (e.g., username and password – not as strong but often used) if a client does not have a suitable certificate, for strong authentication or forgery prevention, server implementations should not allow the connection to be established. In the event that a client does not have a certificate or an acceptable certificate, the server should terminate that connection with a fatal "handshake failure" alert.

The TLS installations almost always require that servers use a certificate and their private key to authenticate themselves to clients. TLS client authentication is optional, and requires that the client have a suitable certificate, issued by a certification authority accepted by the server.

[24] TLS has standardized AES cipher suites [RFC3268], however, it is expected that these cipher suites are not yet widely supported by commercial products. As products begin to provide support for AES, server implementations should support AES cipher suites as the highest priority cipher suites.

In the most common use of TLS, only the server is authenticated through the TLS protocol itself. The client is assured that:

- the server has a certificate issued by a CA accepted by the client (as a practical matter this means a certificate issued by a CA with a self-signed root certificate that is in the in the client's trusted certificate store);
- the server controls the private key corresponding to the public key in the server's certificate, and;
- that the server's network address is consistent with the address in the server's certificate.

Browser clients typically display a "lock" symbol to inform the client that a secure, "https" session is in effect. The TLS session is frequently used primarily to protect a user password from eavesdroppers, and, in these cases, the authentication of the client to the server application depends entirely on the password, not the TLS protocol. Unless clients carefully manage the contents of the trusted certificate store and check the URI displayed in their browser's windows, it may be possible for a sophisticated "man-in-the-middle" attacker to successfully impersonate a web server and intercept protected data, including passwords.

Where strong cryptographic client authentication is required, the server should use the TLS protocol client authentication option to request a client certificate and use that certificate to cryptographically authenticate the client. The server can also provide the client with a list of the CAs it recognizes. In a successful client-authenticated TLS session both parties are assured that:

- the other party has a certificate issued by an acceptable CA (as a practical matter this means a certificate issued by a CA with a self-signed root certificate (possibly through intermediate CAs) that is in a trusted certificate store), and;
- the other party controls the private key corresponding to the public key in its certificate

In addition, the client knows that the server's network address is consistent with the address in the server's certificate.

When client authentication is used, both parties are strongly authenticated, and all data transferred in the TLS session is protected and bound to the authentication by symmetric keys generated in the authentication process. Unless clients carefully manage the contents of the trusted certificate store and check the URI displayed in their browser's windows, it may be possible for a sophisticated "man-in-the-middle" attacker to successfully impersonate a web server and intercept protected data. However, with client authenticated TLS, it is not possible for the attacker to learn the client's password or private key.[25]

[25] When TLS client authentication is used, there is no client password used, and the private key is never transmitted nor divulged.

Session Resumption

During the initial handshake between the client and server, the server generates a session id and passes this value to the client in the "ServerHello" message. The session id (along with the key material and cipher suite) is stored for later use after completion of the handshake. If the server is willing to resume a session at the request of a client (resubmission of "ClientHello"), the server responds with the original session id and cipher suite in the "ServerHello" message. In the event the server is unwilling to resume the session, the server generates and responds with a new session id.

Typical server implementations are agreeable to resuming a previous session. This is a secure mode of operation as the session keys (along with the pre-master secret and master secret) are known only to the client and server, and are coupled with the initial client authentication to provide the necessary security. If there is a requirement to authenticate each client as they initiate a connection session, the server should be configured to ignore requests to resume a session, and generate a new session id, which forces the entire handshake procedure (including client authentication) to proceed.

5.3 Generation of Random Numbers

Of particular concern to both the client and server is the generation of quality random numbers. Random numbers are used for the generation of keys, and for the generation of any random number that is needed to complete cryptographic exchanges. As such, it is important to strive for the following principles when generating random numbers:

- All possible outputs should occur with equal probability, and a series of outputs should appear to conform to a uniform distribution.
- Given a sequence of output bits, it should not be feasible to compute or predict any other (past or future) output bit.

Random numbers can be obtained using either a non-deterministic random bit generator (NRBG) or a deterministic (pseudorandom) random bit generator (DRBG). NRBGs and DRBGs produce bit strings from which random numbers can be determined. Both types of generators present implementation problems. DRBGs must be properly seeded with random data that should normally be kept secret, and the generator must provide the capability of generating a very large stream of bits without repeating. NRBGs rely on some unpredictable, physical source of randomness that is outside human control to produce random output. Typically, an NRBG may not produce random bits quickly enough. A simple solution to the problem of producing random numbers quickly is to use an NRBG to seed a DRBG. Care must be taken in the design and use of either type of generator to ensure that the requirements for randomness are met. Therefore, random numbers should be generated in modules that are validated under the Cryptographic Module Validation Program (CMVP)[26].

---

[26] The Cryptographic Module Validation Program (CMVP) is a joint venture of NIST and the Communications Security Establishment (CSE) of the Government of Canada. The CMVP validates commercial products for conformance to FIPS 140-1 ([FIPS140-1]) and FIPS 140-2 ([FIPS140-2]), allowing vendors to build to a common standard and utilize one common validation process. Additional information on the CMVP is available at Hhttp://cs-www.ncsl.nist.gov/cryptval/cmvp.htmH.

Note: This is an effort underway within the American National Standards Institute (ANSI) to develop a Random Number Generator standard. NIST plans are to use this as a basis for a Federal standard.

5.4 Operational Considerations

5.4.1 Implementation Considerations

Sections 5.1 and 5.2 of this document provide recommendations for the cipher suites that the clients and browsers should implement for transport layer security within the TLS protocol. System administrators need to fully understand the ramifications of selecting cipher suites and configuring applications to support only those cipher suites. Through current NIST research into products supporting TLS, it was determined that:

- The set of allowed cipher suites for most browsers includes, by default, RC4 for encryption with a 40 bit key,
- There was a limited choice of cipher suites for browsers and servers, and cipher suites with RC4 were typically chosen first,
- Most server implementations do not allow the server administrator to specify preference order. The only way to ensure that a server uses 3DES for encryption, was to configure the server to not implement cipher suites with RC4, and
- On many systems the selection of a cryptographic algorithm was system-wide and not application specific (e.g., disabling an algorithm for one application would disable that algorithm for all applications on Microsoft Windows systems using Microsoft's Cryptographic API).

Of equal importance is the need to specify the key lengths used in the cipher suites (for both clients and servers). Currently, most browsers do not allow for user specification of key lengths, but those utilizing OpenSSL libraries can select either 512 or 1024 bit RSA key sizes. The use of RSA/DSA server X.509 certificates with a minimum 1024 bit key size is acceptable until the year 2010. If the server identity must be authenticated after the year 2010, key sizes larger than 1024 bits are needed.

5.4.2 Additional Operational Concerns

The Hypertext Transfer Protocol (HTTP) is an extremely flexible protocol that allows for many uses and implementations. This flexibility, however, also introduces vulnerabilities that are not and cannot be mitigated solely by Transport Layer Security. Given the ease at which connections to a known server can be hijacked, it is incumbent upon the client to not only check all data received but also verify the pathway of the message and the message's integrity. This includes verifying the server's identity presented in the server's certificate at the time the connection is established.

Likewise, both the server and the client should not base authentication decisions solely upon the Transport Layer Security's mechanism for determining possession of the private

key that corresponds to the exchanged certificate. Rather, the decision should also consider whether or not the certificate is valid or has been revoked.

6. References

The following list of documents, publications, and organizations provide a wide variety of information on varying aspects of Transport Layer Security.

[Adams99] Adams, C. and Lloyd, S., *Understanding PKI: Concepts, Standard, and Deployment Considerations*, (Macmillan Technology Publishing, Indianapolis, IN, ISBN 1-57870-166-X, 1999).

[Comer00] Comer, D. E., *Internetworking with TCP/IP, Principles, Protocols, and Architectures*, Fourth Edition, (Prentice Hall, Upper Saddle River, NJ 07458, ISBN: 0-13- 018380-6, 2000).

[ECCTLS] Gupta, V. *ECC Cipher Suites for TLS*. Draft Internet Engineering Task Force, Request for Comment, October 2004. http://www.ietf.org/proceedings/04aug/I-D/draft-ietf-tls-ecc-06.txt

[Fanto2002] Fanto, M, SSL Web Server and Client Configuration, PKI Technical Working Group Presentation. http://csrc.nist.gov/pki/twg/y2002/presentations/twg-02-16.pdf

[FIPS46-3] FIPS 46-3, *Data Encryption Standard (DES)*[27], http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

[FIPS81A] National Institute of Standards and Technology, *DES Modes of Operation,* Federal Information Processing Standard 81, 1980 December 2, http://csrc.nist.gov/publications/fips/fips81/fips81.htm

[FIPS81B] National Institute of Standards and Technology, *DES Modes of Operation Change Notice 2,* Federal Information Processing Standard 81, 1996 May31, http://csrc.nist.gov/publications/fips/fips81/fips81change2.pdf

[FIPS81C] National Institute of Standards and Technology, *DES Modes of Operation Change Notice 3,* Federal Information Processing Standard 81, http://csrc.nist.gov/publications/fips/fips81/fips81change3.pdf

[FIPS140-1] FIPS 140-1, *Security Requirements For Cryptographic Modules*, http://csrc.nist.gov/publications/fips/fips140-1/fips1401.pdf

[FIPS140-2] FIPS 140-2, *Security Requirements For Cryptographic Modules,* http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

[FIPS140Impl] National Institute of Standards and Technology, *Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program*, April 28, 2004, http://csrc.ncsl.nist.gov/cryptval/140-1/FIPS1402IG.pdf

[FIPS180-2] National Institute of Standards and Technology, *Secure Hash Standard (+ Change Notice to include SHA-224)*, Federal Information Processing Standards Publication 180-2, August 1 2002, http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf

[27] FIPS 46-3 is in the process of being withdrawn and replaced by NIST Special Publication 800-67, currently available at Hhttp://csrc.nist.gov/publications/nistpubs/800-67/SP800-67.pdfH

[FIPS186-2] National Institute of Standards and Technology, *Digital Signature Standard*[28], Federal Information Processing Standard 186-2, 27 January 2000, http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf.

[FIPS197] National Institute of Standards and Technology, *Advanced Encryption Standard (AES),* Federal Information Processing Standard 197, November 26, 2001 http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[FIPS198] National Institute of Standards and Technology, *The Keyed-Hash Message Authentication Code (HMAC)*, Federal Information Processing Standard 198, 6 March 2002, http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf.

[Hall00] Hall, E. A., *Internet Core Protocols, The Definitive Guide*, (O'Reilly & Associates, ISBN: 1-56592-572-6, February 2000).

[Hollenbeck04] Hollenbeck, S., *Transport Layer Security Protocol Compression Methods*, Internet Engineering Task Force, Internet Draft, 16 January 2004. Transport Layer Security Protocol Compression Methods, http://www.ietf.org/internet-drafts/draft-ietf-tls-compression-07.txt (expires July 16, 2004). (Note: See http://www.ietf.org/html.charters/tls-charter.html for updated drafts and status.)

[Housley01] Housley, R. and Polk, T., *Planning for PKI, Best Practices Guide for Deploying Public Key Infrastructure*, (John Wiley & Sons, New York, NY, ISBN 0-471-39702-4, 2001).

[KeyMgmt03] National Institute of Standards and Technology's Computer Security Resource Center, *Key Management Information,* http://csrc.ncsl.nist.gov/CryptoToolkit/tkkeymgmt.html.

[PKCS1] RSA Laboratories, *PKCS #1 v2.1: RSA Cryptography Standard*, 14 June 2002.

[Polk03] Polk, W., Hastings, N., and Malani, A., *Public Key Infrastructures that Satisfy Security Goals,* IEEE Internet Computing, Volume 7, Number 4, July-August, 2003.

[Rescorla01] Rescorla, E., *SSL and TLS – Designing and Building Secure Systems*, (Addison- Wesley, Upper Saddle River NJ, 07458, ISBN 0-201-61598, March 2001).

[RFC2246] Dierks, T. and Allen, C., *The TLS Protocol Version 1.0*, Internet Engineering Task Force, Request for Comment 2246, January 1999, http://www.ietf.org/rfc/rfc2246.txt

[RFC3268] Chown, P., *Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS),* Internet Engineering Task Force, Request for Comment 3268, June 2002, http://www.ietf.org/rfc/rfc3268.txt

[SP800-23] NIST Special Publication 800-23, *Guidelines to Federal Organizations on Security Assurance and Acquisition/Use of Tested/Evaluated Products,* August 2000, http://csrc.nist.gov/publications/nistpubs/800-23/sp800-23.pdf.

[28] FIPS 186-2 will be replaced by FIPS 186-3, which is currently out for comment.

[SP800-32] NIST Special Publication 800-32, *Introduction to Public Key Technology and the Federal PKI Infrastructure,* February 2001, http://csrc.nist.gov/publications/nistpubs/800-32/sp800-32.pdf

[SP800-36] NIST Special Publication 800-36, *Guide to Selecting Information Technology Security Products,* October 2003, http://csrc.nist.gov/publications/nistpubs/800-36/NIST-SP800-36.pdf.

[Stallings98] Stallings, W., *SSL: Foundation for Web Security*, Internet Protocol Journal, Volume1, Number 1, June 1998, http://www.cisco.com/en/US/about/ac123/ac147/archived_issues/ipj_1-1/ssl.htmlT

[X9.31] American National Standards Institute, *Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (Appendix A.2.4, Generating Pseudorandom Numbers) X9.31*, 1998.

[7498] ISO/IEC 7498-1: 1994(E), ITU-T Rec. X.200 (1994 E), Information Processing Systems - OSI Reference Model - The Basic Model.

NIST
National Institute of
Standards and Technology
U.S. Department of Commerce

Special Publication 800-111

Guide to Storage Encryption Technologies for End User Devices

Recommendations of the National Institute of Standards and Technology

Karen Scarfone
Murugiah Souppaya
Matt Sexton

NIST Special Publication 800-111

Guide to Storage Encryption Technologies for End User Devices

Recommendations of the National Institute of Standards and Technology

Karen Scarfone
Murugiah Souppaya
Matt Sexton

C O M P U T E R S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

November 2007

U.S. Department of Commerce
Carlos M. Gutierrez, Secretary
National Institute of Standards and Technology
James M. Turner, Acting Director

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive unclassified information in Federal computer systems. This Special Publication 800-series reports on ITL's research, guidance, and outreach efforts in computer security and its collaborative activities with industry, government, and academic organizations.

National Institute of Standards and Technology Special Publication 800-111
Natl. Inst. Stand. Technol. Spec. Publ. 800-111, 40 pages (Nov. 2007)

Acknowledgements

The authors, Karen Scarfone and Murugiah Souppaya of the National Institute of Standards and Technology (NIST), and Matt Sexton of Booz Allen Hamilton, wish to thank their colleagues who reviewed drafts of this document and contributed to its technical content. In particular, their appreciation goes to Tim Grance, Bill Burr, and Tim Polk of NIST, and Derrick Dicoi, Angela Orebaugh, Manuel Villar, Mike Zeberlein, and Mike Zirkle of Booz Allen Hamilton, for their keen and insightful assistance throughout the development of the document. The authors also appreciate the efforts of those individuals, agencies, and other organizations that contributed input during the public comment period.

Table of Contents

List of Tables

Executive Summary

In today's computing environment, there are many threats to the confidentiality of information stored on end user devices, such as personal computers, consumer devices (e.g., personal digital assistant, smart phone), and removable storage media (e.g., universal serial bus [USB] flash drive, memory card, external hard drive, writeable CD or DVD). Some threats are unintentional, such as human error, while others are intentional. Intentional threats are posed by people with many different motivations, including causing mischief and disruption and committing identity theft and other fraud. A common threat against end user devices is device loss or theft. Someone with physical access to a device has many options for attempting to view or copy the information stored on the device. Another concern is insider attacks, such as an employee attempting to access sensitive information stored on another employee's device. Malware, another common threat, can give attackers unauthorized access to a device, transfer information from the device to an attacker's system, and perform other actions that jeopardize the confidentiality of the information on a device.

Many threats against end user devices could cause information stored on the devices to be accessed by unauthorized parties. To prevent such disclosures of information, particularly of personally identifiable information (PII) and other sensitive data, the information needs to be secured. Securing other components of end user devices, such as operating systems, is also necessary, but in many cases additional measures are needed to secure the stored information. This publication explains the basics of storage security, which is the process of allowing only authorized parties to access and use stored information. The primary security controls for restricting access to sensitive information stored on end user devices are encryption and authentication. Encryption can be applied granularly, such as to an individual file containing sensitive information, or broadly, such as encrypting all stored data. The appropriate encryption solution for a particular situation depends primarily upon the type of storage, the amount of information that needs to be protected, the environments where the storage will be located, and the threats that need to be mitigated. This publication describes three types of solutions—full disk encryption, volume and virtual disk encryption, and file/folder encryption—and makes recommendations for implementing and using each type. This publication also includes several use case examples, which illustrate that there are multiple ways to meet most storage encryption needs. When selecting a solution type, organizations should consider the range of solutions that meet the organization's security requirements, and not just the solution type that is most commonly used.

Implementing the following recommendations should facilitate more efficient and effective storage encryption solution design, implementation, and management for Federal departments and agencies.

When selecting a storage encryption technology, organizations should consider solutions that use existing system features (such as operating system features) and infrastructure.

There are many factors for organizations to consider when selecting storage encryption solutions, such as the platforms they support, the data they protect, and the threats they mitigate. Some solutions involve deploying various servers and installing software on the devices to be protected, while other solutions can use existing servers, as well as software built into the devices to be protected, such as Federal Information Processing Standard (FIPS) approved encryption features built into the devices' operating systems. Generally, the more extensive the changes are to the infrastructure and devices, the more likely it is that the storage encryption solution will cause a loss of functionality or other problems with the devices. When evaluating solutions, organizations should compare the loss of functionality with the gain in security capabilities and decide if the tradeoff is acceptable. Solutions that require extensive changes to the infrastructure and end user devices should generally be used only when other solutions cannot meet the organization's needs.

Organizations should use centralized management for all deployments of storage encryption except for standalone deployments and very small-scale deployments.

Centralized management is recommended for most storage encryption deployments because of its effectiveness and efficiency for policy verification and enforcement, key management, authenticator management, data recovery, and other management tasks. Centralized management can also automate deployment and configuration of storage encryption software to end user devices, distribution and installation of updates, collection and review of logs, and recovery of information from local failures.

Organizations should ensure that all cryptographic keys used in a storage encryption solution are secured and managed properly to support the security of the solution.

Storage encryption technologies use one or more cryptographic keys to encrypt and decrypt the data that they protect. If a key is lost or damaged, it may not be possible to recover the encrypted data from the computer. Therefore, organizations should perform extensive planning of key management processes, procedures, and technologies before implementing storage encryption technologies. This planning should include all aspects of key management, including key generation, use, storage, recovery, and destruction. Organizations should carefully consider how key management practices can support the recovery of encrypted data if a key is inadvertently destroyed or otherwise becomes unavailable. Organizations planning on encrypting removable media also need to consider how changing keys will affect access to encrypted storage on removable media and develop feasible solutions, such as retaining the previous keys in case they are needed.

Organizations should select appropriate user authenticators for storage encryption solutions.

Storage encryption solutions require users to authenticate successfully before accessing the information that has been encrypted. Common authentication mechanisms are passwords, personal identification numbers, cryptographic tokens, biometrics, and smart cards. Organizations should consider leveraging existing enterprise authentication solutions (e.g., Active Directory, public key infrastructure [PKI]) instead of adding another authenticator for users. Generally, this is acceptable if two-factor authentication is being used. Using the same single-factor authenticator for multiple purposes, such as operating system (OS) authentication and storage encryption authentication, significantly weakens the protection that authentication provides. For example, an attacker who learns a single password could gain full access to the device's information. Organizations should carefully consider the security implications of using the same single-factor authenticator for multiple purposes. In particular, organizations should not use email passwords and other passwords sometimes transmitted in plaintext as single-factor authenticators for storage encryption.

Organizations should implement measures that support and complement storage encryption implementations for end user devices.

Storage encryption by itself cannot provide adequate security for stored information; additional security controls are needed. Organizations should select and deploy the necessary controls based on FIPS 199's categories for the potential impact of a security breach involving a particular system and NIST Special Publication 800-53's recommendations for minimum management, operational, and technical security controls. Examples of supporting controls are as follows:

- Revising organizational policies as needed to incorporate appropriate usage of the storage encryption solution.

- Securing and maintaining end user devices properly, which should reduce the risk of compromise or misuse. This includes securing device operating systems, applications, and communications, and physically securing devices.
- Making users aware of their responsibilities for storage encryption, such as encrypting sensitive files, physically protecting mobile devices and removable media, and promptly reporting loss or theft of devices and media.

1. Introduction

1.1 Authority

The National Institute of Standards and Technology (NIST) developed this document in furtherance of its statutory responsibilities under the Federal Information Security Management Act (FISMA) of 2002, Public Law 107-347.

NIST is responsible for developing standards and guidelines, including minimum requirements, for providing adequate information security for all agency operations and assets; but such standards and guidelines shall not apply to national security systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), "Securing Agency Information Systems," as analyzed in A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided in A-130, Appendix III.

This guideline has been prepared for use by Federal agencies. It may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright, though attribution is desired.

Nothing in this document should be taken to contradict standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority, nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official.

1.2 Purpose and Scope

The purpose of this document is to assist organizations in understanding storage encryption technologies for end user devices and in planning, implementing, and maintaining storage encryption solutions. The types of end user devices addressed in this document are personal computers (desktops and laptops), consumer devices (e.g., personal digital assistants, smart phones), and removable storage media (e.g., USB flash drives, memory cards, external hard drives, writeable CDs and DVDs). This document provides practical, real-world guidance for three classes of storage encryption techniques: full disk encryption, volume and virtual disk encryption, and file/folder encryption. It also discusses important security elements of a storage encryption deployment, including cryptographic key management and authentication. It only discusses the encryption of data at rest (storage), and does not address the encryption of data in motion (transmission).

1.3 Audience

This document has been created for information security program managers and staff, system administrators, and others who are responsible for selecting, deploying, managing, and maintaining storage encryption technologies for end user devices. This document does not assume that the reader has previous experience with any storage encryption technologies, but it does assume that the reader has experience with information security.

1.4 Document Structure

The remainder of this document is organized into three major sections.

- Section 2 provides an overview of the basic concepts of storage encryption for end user devices.

- Section 3 describes the most commonly used categories of storage encryption technologies for end user devices, and explains the types of protection they provide.
- Section 4 discusses the process of planning and implementing storage encryption technologies for end user devices. It includes a detailed discussion of the importance of cryptography and authentication to a storage encryption solution.

The document also contains several appendices with supporting material.

- Appendix A describes alternatives to encrypting storage on end user devices.
- Appendices B and C contain a glossary and acronym list, respectively.
- Appendix D lists online tools and resources that may be useful references for gaining a better understanding of storage encryption for end user devices.

2. Storage Security Overview

An *end user device* is a personal computer (desktop or laptop), consumer device (e.g., personal digital assistant [PDA], smart phone), or removable storage media (e.g., USB flash drive, memory card, external hard drive, writeable CD or DVD) that can store information.[1] *Storage security* is the process of allowing only authorized parties to access and use stored information. This section introduces the basic concepts of storage security for end user devices.[2]

2.1 File Storage Basics

A *file* is a collection of information logically grouped into a single entity and referenced by a unique name, such as a *filename*. On end user devices, there are typically two types of files: data files, such as text documents, spreadsheets, images, and videos, and system files, such as operating system and application binaries and libraries. A *filesystem* defines the way that files are named, stored, organized, and accessed. *Directories*, also known as *folders*, are organizational structures used by filesystems to group files. Another feature of a filesystem is *metadata*, which is data about data—in the context of a filesystem, information regarding the files and folders themselves, such as file and folder names, creation dates and times, and sizes.

Filesystems are designed to store folders, system and data files, and metadata on storage media. However, storage media may also hold *residual data*, which is data from deleted files (including earlier versions of existing files and temporary files). Residual data can often be recovered from an end user device through forensic analysis. The following items describe common forms of residual data:

- **Unused File Allocation Units.** Filesystems store files in chunks known as file allocation units. *Unused file allocation units* are the units within a partition that are not currently being used by the filesystem. When a file is deleted, it is typically not erased from the media; instead, the information in the directory's data structure that points to the location of the file is marked as deleted. This means that the file is still stored on the media but is no longer enumerated by the operating system (OS). The OS considers this to be unused space and can overwrite any portion of or the entire deleted file at any time.
- **Slack Space.** Even if a file requires less space than the file allocation unit size, an entire file allocation unit is still reserved for the file. For example, if the file allocation unit size is 32 kilobytes (KB) and a file is only 7 KB, the entire 32 KB is still allocated to the file, but only 7 KB is used, resulting in 25 KB of unused space. This unused space is referred to as *file slack space,* and it may hold residual data such as portions of deleted files.
- **Free Space.** *Free space* is the area on media that is not currently allocated to a partition. This often includes space on the media where files may have resided at one point but have since been deleted. The free space may still contain pieces of data.

Before media can be used to store files, the media must usually be partitioned and formatted into logical volumes. *Partitioning* is the act of logically dividing a media into portions that function as separate units. A *logical volume* is a partition or a collection of partitions acting as a single entity that has been formatted

---

[1] This publication only addresses technologies for encrypting files stored on end user devices. Information on storage security for servers, storage area networks, enterprise backup tapes, and other devices is outside the scope of this publication.

[2] Storage security is only one component of data security, which includes network, host, and application security, and also addresses how data may be used after it is accessed. All elements of data security other than storage security, such as encrypting data in motion (e.g., network communications), are outside the scope of this publication.

with a filesystem. Some media types can contain only one partition (and consequently, one logical volume), while others can contain multiple partitions.

2.2 The Need for Storage Security

In today's computing environment, there are many threats to the confidentiality of information stored on end user devices. Some threats are unintentional, such as human error, while others are intentional. Intentional threats are posed by people with many different motivations, including causing mischief and disruption and committing identity theft and other fraud. One of the most common threats is *malware*, also known as *malicious code*, which refers to a program that is inserted into a system, usually covertly, with the intent of compromising the confidentiality, integrity, or availability of the victim's data, applications, or OS. Types of malware threats include viruses, worms, malicious mobile code, Trojan horses, rootkits, and spyware. Malware can give attackers unauthorized access to a device, transfer information from the device to an attacker's system, and perform other actions that jeopardize the confidentiality of the information on a device. Another common threat against end user devices is device loss or theft. Someone with physical access to a device has many options for attempting to view the information stored on the device. This is also a concern for insider attacks, such as an employee attempting to access sensitive information stored on another employee's device. Another form of insider attack is a user attempting to access another user's files on a device that the two users share.

Many threats against end user devices could cause information stored on the devices to be accessed by unauthorized parties. To prevent such disclosures of information, particularly of personally identifiable information (PII)[3] and other sensitive data, the information needs to be secured. Securing other components of end user devices, such as OSs, is also necessary, but in many cases additional measures are needed to secure the stored information. For example, without these additional measures, an attacker that steals a device could use forensic tools and techniques to recover information directly from the storage media, circumventing the protections applied by the device's OS.

A number of laws and regulations compel organizations to ensure that sensitive information is protected appropriately. The following is a list of key regulations, standards, and guidelines that help define organizations' needs for storage security:[4]

- **Federal Information Security Management Act of 2002 (FISMA).** FISMA emphasizes the need for each Federal agency to develop, document, and implement an organization-wide program to provide information security for the information systems that support its operations and assets. NIST Special Publication (SP) 800-53, *Recommended Security Controls for Federal Information Systems*, was developed in support of FISMA.[5] NIST SP 800-53 is the primary source of recommended security controls for Federal agencies. It describes several controls related to storage security, such as controlling access through encryption of stored information, restricting access to mobile computing devices and information system media, and storing media in physically secure locations.

---

[3] OMB Memorandum 06-19, "Reporting Incidents Involving Personally Identifiable Information and Incorporating the Cost for Security in Agency Information Technology Investments", defines PII as "any information about an individual maintained by an agency, including, but not limited to, education, financial transactions, medical history, and criminal or employment history and information which can be used to distinguish or trace an individual's identity, such as their name, social security number, date and place of birth, mother's maiden name, biometric records, etc., including any other personal information which is linked or linkable to an individual." The full text of the memorandum is available at http://www.whitehouse.gov/omb/memoranda/fy2006/m-06-19.pdf.

[4] It is outside this publication's scope to explain which types of information organizations need to protect and how each type should be protected. NIST SP 800-60, *Guide for Mapping Types of Information and Information Systems to Security Categories*, discusses the identification of common types of information. It is at http://csrc.nist.gov/publications/nistpubs/.

[5] Copies of FISMA and NIST SP 800-53 are available at http://csrc.nist.gov/sec-cert/ca-library.html.

- **OMB Memorandum M-06-16.** OMB has issued a memorandum directly related to storage security. OMB M-06-16 addresses the protection of agency information that is either "accessed remotely or physically transported outside of the agency's secured, physical perimeter". It specifically requires that agencies encrypt all data stored on mobile computing devices, such as laptops and PDAs, unless the data has been determined by the designated agency official to be non-sensitive.[6] Similar requirements are also included in OMB Memorandum M-07-16.[7]

- **Privacy Act of 1974.** The Privacy Act regulates the collection, use, maintenance, and dissemination of personal information about U.S. citizens or aliens lawfully admitted for permanent residence. It applies to records maintained by agencies in the executive branch of the government.

- **Gramm-Leach-Bliley Act (GLBA).** GLBA requires financial institutions to protect their customers' information against security threats. This includes ensuring "the security and confidentiality of customer records and information" and protecting "against unauthorized access to or use of such records or information".[8]

- **Health Insurance Portability and Accountability Act of 1996 (HIPAA).** HIPAA includes security standards for certain health information. NIST SP 800-66, *An Introductory Resource Guide for Implementing the Health Insurance Portability and Accountability Act (HIPAA) Security Rule*, lists HIPAA-related storage security needs.[9] For example, Section 4.14 of NIST SP 800-66 describes the need to encrypt and decrypt electronic protected health information (EPHI).

2.3 Security Controls for Storage

The primary security controls for restricting access to sensitive information stored on end user devices are encryption and authentication. Encryption can be applied granularly, such as to an individual file containing sensitive information, or broadly, such as encrypting all stored data. The appropriate encryption solution for a particular situation depends primarily upon the type of storage, the amount of information that needs to be protected, the environments where the storage will be located, and the threats that need to be mitigated. Section 3 discusses the most commonly used options in detail; Appendix A briefly discusses some additional options. Storage encryption solutions require users to authenticate successfully before accessing the information that has been encrypted. Common authentication mechanisms are passwords, personal identification numbers (PIN), cryptographic tokens, biometrics, and smart cards. The combination of encryption and authentication helps control access to the stored information.

Organizations also need to consider the security of backups of stored information. Some organizations permit users to back up their local files to a centralized system, while other organizations recommend that their users perform local backups (e.g., burning CDs, external USB storage media). In the latter case, organizations should ensure that the backups will be secured at least as well as the original source. This could be done with similar controls, such as encrypting the backups, or with different types of controls, such as storing backup tapes in a physically secured room within the organization's facilities.

Organizations should also implement other measures that support and complement storage encryption implementations. These measures help to ensure that storage encryption is implemented in an

---

6 The memorandum is available at http://www.whitehouse.gov/OMB/memoranda/fy2006/m06-16.pdf.

7 M-07-16 is available at http://www.whitehouse.gov/omb/memoranda/fy2007/m07-16.pdf.

8 More information on GLBA is available at http://www.ftc.gov/privacy/privacyinitiatives/glbact.html. A copy of GLBA can be downloaded from http://www.ftc.gov/privacy/privacyinitiatives/financial_rule_lr.html.

9 HIPAA is available at http://www.hhs.gov/ocr/hipaa/, and NIST SP 800-66 is available at http://csrc.nist.gov/publications/nistpubs/.

environment with the management, operational, and technical controls necessary to provide adequate security for the storage encryption implementation. Examples of supporting measures are as follows:

- Revise organizational policies as needed to incorporate appropriate usage of the storage encryption solution. Policies should provide the foundation for the planning and implementation of storage encryption.

- Ensure that end user devices are secured and maintained properly, which should reduce the risk of compromise or misuse. This includes securing device OSs, applications, and communications (e.g., encrypting wired and wireless network traffic) and physically securing devices, such as requiring that laptops be secured using cable locks when in hotels, conferences, and other locations where third parties could easily gain physical access to the devices. Physical security for devices is also an important consideration in home environments, such as preventing others within the house from using an organization-issued device by keeping the device in a locked desk or room.

- Make users aware of their responsibilities for storage encryption, such as encrypting sensitive files, physically protecting mobile devices and removable media, and promptly reporting loss or theft of devices and media.

Organizations should select and deploy the necessary security controls based on existing guidelines. Federal Information Processing Standards (FIPS) 199 establishes three security categories—low, moderate, and high—based on the potential impact of a security breach involving a particular system.[10] NIST SP 800-53 provides recommendations for minimum management, operational, and technical security controls for information systems based on the FIPS 199 impact categories.[11] The recommendations in NIST SP 800-53 should be helpful to organizations in identifying controls that are needed to protect end user devices, which should be used in addition to the specific recommendations for storage encryption listed in this document.[12]

---

10 FIPS 199, *Standards for Security Categorization of Federal Information and Information Systems*, is available at http://csrc.nist.gov/publications/fips/.

11 NIST SP 800-53 Revision 1, *Recommended Security Controls for Federal Information Systems*, is available at http://csrc.nist.gov/publications/nistpubs/.

12 In addition to securing the wireless networks, the wireless devices using the networks also need to be secured; however, an explanation of securing laptops, PDAs, and other wireless devices is outside the scope of this guide.

3. Storage Encryption Technologies

There are many technologies available for encrypting data stored on end user devices. This section describes the most commonly used technologies, discusses the protections provided by each type, and explains how these technologies are typically managed.

3.1 Common Types of Storage Encryption Technologies

This section provides a high-level overview of the most commonly used options for encrypting stored information: full disk encryption, volume and virtual disk encryption, and file/folder encryption.[13] It briefly defines each option and explains at a high level how it works.

3.1.1 Full Disk Encryption

Full disk encryption (FDE), also known as whole disk encryption, is the process of encrypting all the data on the hard drive used to boot a computer, including the computer's OS, and permitting access to the data only after successful authentication to the FDE product. Most FDE products are software-based, so this section focuses on explaining the capabilities and characteristics of software-based FDE solutions. Hardware-based solutions are discussed briefly at the end of this section.

FDE software works by redirecting a computer's *master boot record (MBR)*, which is a reserved sector on bootable media that determines which software (e.g., OS, utility) will be executed when the computer boots from the media. Before FDE software is installed onto a computer, the MBR usually points to the computer's primary OS. When FDE software is being used, the computer's MBR is redirected to a special pre-boot environment (PBE) that controls access to the computer.[14] This redirection is depicted in Figure 3-1. The PBE prompts the user to authenticate successfully, such as entering a user ID and password, before decrypting and booting the OS. This is known as *pre-boot authentication (PBA)*.[15] Most FDE products support the use of both network-based authentication (e.g., Active Directory, PKI) and local authentication sources (e.g., locally stored, locally cached from network source) for PBA.

Once successful PBA occurs, the FDE software decrypts the boot sector for the OS, as depicted by the second arrow in Figure 3-1, and the boot loader in the boot sector starts to load the OS. As it loads, the FDE software decrypts the OS files (which are stored in the system volume) as needed, indicated in Figure 3-1 by the third arrow. Once the OS has finished booting, the user provides OS authentication and uses the computer normally. When the user needs to open encrypted files, save new files, or perform other operations involving the hard drive, the FDE software transparently decrypts and encrypts the necessary sectors[16] of the hard drive as needed.[17] This may marginally increase the time needed to open or save files, but the delay generally should only be noticeable for particularly large files. On an FDE-protected computer, users will typically notice a delay of at least a few seconds when booting the

---

13 Information stored on end user devices can be encrypted in many ways. For example, an application that accesses sensitive information could be responsible for encrypting that information. Applications such as backup programs might also offer encryption options. Another method for protecting files is digital rights management (DRM) software.

14 Some FDE implementations verify the integrity of the boot components, including the MBR, before proceeding. Figure 3-1 depicts the PBE as being stored in the space between the MBR and the boot sector; although many software-based FDE products store the PBE there, this is not required, and some products store the PBE elsewhere.

15 Most software-based FDE products use PBA. Other products require the user to authenticate after the OS has booted, which provides weaker protection than PBA. This publication assumes that FDE implementations are configured to require PBA. PBA can be performed with stronger authenticators than user ID and password; see Section 4.2 for additional information.

16 A *sector* is the smallest logical component of a hard drive that can be read or written. Many hard drives have 512-byte sectors. Even if a single byte of data needs to be accessed, the hard drive will read the entire sector containing that data.

17 Figure 3-1 does not show data volumes and other volumes that might be on a hard drive; such volumes would also be encrypted by an FDE solution.

computer or shutting it down. Delays may also occur when using hibernation[18] features, because the FDE software has to encrypt and decrypt the large hibernation file (which includes a copy of the computer's memory) that is stored on the hard drive. The length of delays is dependent on the size of memory, the hard drive's size and speed, and other factors.

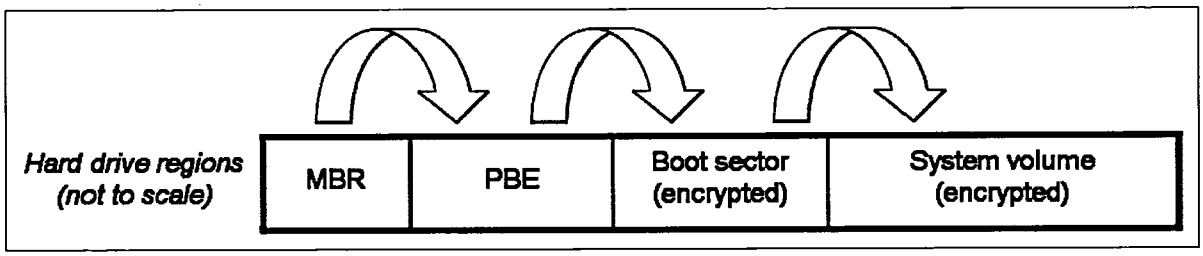

Figure 3-1. Boot Sequence for Full Disk Encryption Software

Because FDE alters how a computer boots, it can cause operational problems. For example, modifying the MBR can prevent computers with dual-boot configurations from functioning properly,[19] and storing the PBE in the space between the MBR and the boot sector can cause conflicts with other software, such as disk-level software tools, that also store code in that space. FDE-protected devices may also have problems with asset management tools and the use of wake-on-LAN.[20]

FDE software is most commonly used on desktop and laptop computers. The requirement for pre-boot authentication means that users have to be able to authenticate using the most fundamental components of a device, such as a standard keyboard—because the OS is not loaded, OS-level drivers are unavailable. For example, a PDA or smart phone could not display a keyboard on the screen for entering a password because that is an OS-level capability.

As mentioned at the beginning of the section, FDE can also be built into a hard drive disk controller.[21] Hardware and software-based FDE offer similar capabilities through different mechanisms. When a user tries to boot a device protected with hardware-based FDE, the hard drive prompts the user to authenticate before it allows an OS to load. The FDE capability is built into the hardware in such a way that it cannot be disabled or removed from the drive. The encryption code and authenticators, such as passwords and cryptographic keys, are stored securely on the hard drive. Because the decryption and encryption is performed by the hard drive itself, with no OS participation, typically there is very little performance impact.

---

[18] Hibernation refers to saving the state of the computer, including the contents of memory, and placing the computer in a low power-usage mode that uses just enough power to maintain its state. Depending on the OS, hibernation mode may also be called sleep, standby, or suspend mode; however, some of these terms do not have universally accepted definitions, and some OSs have features with these names that do not actually write memory out to a file. Modes that do not write memory to a file should not be used with FDE software because the FDE software will not protect the data in these modes.

[19] Dual-boot computers usually modify the MBR to enable users to select which OS will be booted. The details of this depend on the OSs and any utilities used to manage the disk partitions. Using FDE with a dual-boot computer increases the complexity of the configuration and the likelihood for errors that cause loss of data or loss of availability.

[20] The problem with wake-on-LAN technologies is the requirement to perform PBA before booting the system. Some FDE products can be configured to skip PBA when wake-on-LAN is used, either every time or for a certain number of reboots. Access to the OS logon can be suppressed to somewhat compensate for the lack of PBA. However, skipping PBA during wake-on-LAN still introduces additional risk, because an attacker that takes a device that is configured this way could easily set up wake-on-LAN services, circumvent PBA, and use forensic tools to gain access to the device's information. Organizations should carefully consider these risks before using wake-on-LAN for FDE-protected devices.

[21] As of mid-2007, few hardware-based FDE products were yet available, but vendors had announced that several additional products would be available in the coming months.

A major difference between software and hardware-based FDE is that software-based FDE can be centrally managed, but hardware-based FDE can usually only be managed locally. This makes key management and recovery actions considerably more resource-intensive and cumbersome for hardware-based FDE than software-based. Another major difference is that because hardware-based FDE does all cryptographic processing within the hard drive's hardware, it does not need to place its cryptographic keys into the computer's memory, which could potentially expose the keys to malware and other threats. A third significant difference is that hardware-based FDE typically does not alter the MBR, so hardware-based FDE does not cause conflicts with software that modifies the MBR (e.g., dual-boot configurations).

3.1.2 Virtual Disk Encryption and Volume Encryption

Virtual disk encryption is the process of encrypting a file called a *container*, which can hold many files and folders, and permitting access to the data within the container only after proper authentication is provided, at which point the container is typically mounted as a virtual disk. Virtual disk encryption is used on all types of end user device storage. The container is a single file that resides within a logical volume. Examples of volumes are boot, system, and data volumes on a personal computer, and a USB flash drive formatted with a single filesystem. *Volume encryption* is the process of encrypting an entire logical volume and permitting access to the data on the volume only after proper authentication is provided. Volume encryption is most often performed on hard drive data volumes and volume-based removable media, such as USB flash drives and external hard drives. Volume encryption of boot and system volumes is essentially a special form of FDE, and it is not discussed in this section; see the FDE material in Section 3.1.1 for additional information.

At a high level, volume and virtual disk encryption are performed similarly. Software running on the OS used to access the volume or container handles all attempts to read to or write from the protected volume or container.[22] Once the OS has been loaded, if the user needs to use the encrypted volume or container, it will be mounted after the user has provided the required authentication. The software will then automatically decrypt and encrypt the appropriate sectors as needed. This increases the time needed to open or save files, but the delay generally should be noticeable for only particularly large files. There may also be slight delays associated with mounting and unmounting an encrypted volume or container.

The key difference between volume and virtual disk encryption is that containers are portable and volumes are not—a container can be copied from one medium to another, with encryption intact. This allows containers to be burned to CDs and DVDs and to be used on other media that are not volume-based. Virtual disk encryption also makes it trivial to back up sensitive data; the container is simply copied to the backup server or media. Another advantage of virtual disk encryption over volume encryption is that virtual disk encryption can be used in situations where volume-based removable media needs to have both protected and unprotected storage; the volume can be left unprotected and a container placed onto the volume for the sensitive information.

Some virtual disk encryption products further support mobility by offering features that can place executables on the medium holding a container. The medium can then be moved to another computer and the executables run, through methods such as installing drivers onto the computer or running an authentication and decryption utility.[23] The protected contents of the medium can then be accessed by a user after providing the requested authentication.

The responsibilities of the users of virtual disk and volume encryption solutions vary, primarily depending on the devices' access control. For example, if a laptop's OS is configured so that a user can

[22] Some products install kernel mode drivers to perform volume and virtual disk encryption. Other products, especially those specifically designed for removable media, either contain their own resident OSs or provide software applications.

[23] This only speaks to the portability of the logical entity on the media—the media itself might be physically portable.

only write files to an encrypted container or volume, then the user does not need to take steps to ensure that files are saved to the appropriate location. However, if the OS is not configured this way, permitting users to save files to various locations, or if the encrypted device is removable media that is not protected through OS access control features, then users will be responsible for ensuring that they save files in the appropriate location. In this case, if users fail to follow the necessary procedures, then some files that should be protected may not be.

3.1.3 File/Folder Encryption

File encryption is the process of encrypting individual files on a storage medium and permitting access to the encrypted data only after proper authentication is provided. *Folder encryption* is very similar to file encryption, only it addresses individual folders instead of files. Some OSs offer built-in file and/or folder encryption capabilities,[24] and many third-party programs are also available. Although folder encryption and virtual disk encryption sound similar—both a folder and a container are intended to contain and protect multiple files—there is a difference. A container is a single opaque file, meaning that no one can see what files or folders are inside the container until the container is decrypted. File/folder encryption is transparent, meaning that anyone with access to the filesystem can view the names and possibly other metadata for the encrypted files and folders, including files and folders within encrypted folders, if they are not protected through OS access control features. File/folder encryption is used on all types of storage for end user devices.

File/folder encryption can be implemented in many ways, including through drivers, services, and applications. When a user attempts to open an encrypted file (either encrypted by itself or located in an encrypted folder), the software requires the user to first authenticate successfully. Once that has been done, the software will automatically decrypt the chosen file. Because it decrypts a single file at a time, the performance impact of file/folder encryption should be minimal. File/folder encryption is most commonly used on user data files, such as word processing documents and spreadsheets. File/folder encryption solutions can sometimes encrypt swap files, but typically not OS executables and hibernation files.

Many file/folder encryption products offer several options for selecting which files and folders should be encrypted and defining the user's role in using the solution—manually enabling encryption for each new file or folder that needs protection, remembering to store files and folders in the proper locations, or doing nothing differently because the files and folders are encrypted automatically. Common options include:

- Relying on the user to specifically designate the files and folders
- Automatically encrypting the contents of administrator-designated folders
- Automatically encrypting certain types of files, such as those with a particular file extension
- Automatically encrypting all files written to by particular applications
- Automatically encrypting all data files for particular users.

There are also various applications, such as file compression utilities and office productivity suites, that offer limited file/folder encryption capabilities. Such applications are usually completely dependent on the user to ensure that the necessary files are encrypted, and these applications are often not centrally managed, which can complicate key management and other aspects of managing the use of the

[24] For example, New Technology File System (NTFS) supports file and folder encryption using the Encrypting File System (EFS).

applications' file/folder encryption features. Appendix A presents additional examples of applications that can encrypt the information that they store.

3.2 Protection Provided by Storage Encryption Technologies

The following explains the types of protection each storage encryption technology can and cannot provide.

- **Full Disk Encryption.** For a computer that is not booted, all the information encrypted by FDE is protected, assuming that pre-boot authentication is required. When the device is booted, then FDE provides no protection; once the OS is loaded, the OS becomes fully responsible for protecting the unencrypted information. The exception to this is when the device is in a hibernation mode; most FDE products can encrypt the hibernation file.

- **Virtual Disk and Volume Encryption.** When virtual disk encryption is employed, the contents of containers are protected until the user is authenticated for the containers. If single sign-on is being used for authentication to the solution, this usually means that the containers are protected until the user logs onto the device. If single sign-on is not being used, then protection is typically provided until the user explicitly authenticates to a container. Virtual disk encryption does not provide any protection for data outside the container, including swap and hibernation files that could contain the contents of unencrypted files that were being held in memory. Volume encryption provides the same protection as virtual disk encryption, but for a volume instead of a container.

- **File/Folder Encryption.** File/folder encryption protects the contents of encrypted files (including files in encrypted folders) until the user is authenticated for the files or folders. If single sign-on is being used, this usually means that the files are only protected until the user logs onto the device. If single sign-on is not being used, then protection is typically provided until the user explicitly authenticates to a file or folder. File/folder encryption does not provide any protection for data outside the protected files or folders, including swap and hibernation files that could contain the contents of unencrypted files that were being held in memory. File/folder encryption software also cannot protect the confidentiality of filenames and other file metadata, which itself could provide valuable information to attackers (for examples, files that are named by Social Security number).

In many cases, especially for FDE and volume encryption, these products do not provide any protection for files copied or moved from the encrypted storage to another location (either local or on the network), because they automatically decrypt the files as part of the copy or move process.[25] The target location is responsible for protecting the files, and no protection is provided in transit from the source to the target. However, some storage encryption technologies allow protection to be retained if desired. Most virtual disk encryption products allow an entire container to be transferred, including the container's protection, but individual files or folders copied or moved from a container will not be protected. Some file/folder encryption products allow files or folders to retain their protection when they are copied or moved, in some cases only within a single filesystem, and in other cases both within a single filesystem and to other filesystems.

The main threat that all these types of technology mitigate is unauthorized access to information on a lost or stolen device. Virtual disk/volume encryption and file/folder encryption technologies can also mitigate some OS and application layer threats to protected information involving malware, remote access to the protected information, and other methods that depend on the OS being booted, until the user successfully

[25] Some products display a warning message or prompt the user to confirm the action before decrypting and copying or moving the files.

authenticates to the encryption solution. Once this authentication occurs, then any process being run on the device (such as malware) with access to the user's files can get the decrypted information. Because the files are only protected until successful authentication occurs, it may be beneficial to use a solution that is configured to encrypt only the necessary files (e.g., using file/folder encryption to encrypt 10 sensitive files instead of using volume encryption to encrypt 10 sensitive files and 1000 non-sensitive files). The more files that are protected, the sooner the user is likely to authenticate to the storage encryption solution, which increases the window of exposure for the decrypted files.

Some products also permit storage to be encrypted either for a single user or for multiple users of a device. If encrypted for a single user, the confidentiality of that user's encrypted storage is protected from other users of the device, including (in most cases) the device's administrators. Encrypting for multiple users allows sensitive data to be shared by those users, while still protecting it from other users of the device. This provides some protection against insider threats.

In some cases, multiple types of technology can be used concurrently to protect against different classes of threats; for example, FDE could be used to protect all data on a device from device loss or theft, and volume, virtual disk, or file/folder encryption could be used to provide additional protection for a subset of data that is more sensitive than the rest of the data.[26]

When thinking about threats, organizations should be aware that after storage encryption technology has been implemented, there may be residual data on the device that remains unprotected. For example, when a file is encrypted using file/folder encryption and the original file is deleted, the remnants of the original plaintext file might still be present on the storage media. Another example is FDE and volume encryption products that encrypt only the disk sectors that contain current files, not disk sectors that only contain deleted files or other remnants of data. These remnants may be recoverable using forensic tools by an attacker who gets physical access to the computer, without having to provide any authentication. Organizations should take into account threats against both the files and remnants of the files.

Organizations should be aware that if an end user device is compromised at any time, any storage encryption technologies on it may become partially or wholly ineffective. For example, when the device is in use and the user has been authenticated to the storage encryption solution, malware could access decrypted files and transfer copies of them to external hosts or extract sensitive information from them. Other examples are an attacker disabling or reconfiguring storage encryption, malware installing a keylogger that captures passwords used for storage encryption authentication, or malware acquiring a copy of a storage encryption key from the device's memory (for software-based storage encryption solutions).

Organizations should also be aware that they should not rely on storage encryption technologies to protect data without regularly maintaining the encryption solution. For example, if an attacker acquires a lost, stolen, or retired device protected by storage encryption technology, and a vulnerability in the storage encryption technology is discovered in the future, the attacker may be able to exploit it to access the protected data.

3.3 Comparison of Storage Encryption Technologies

Table 3-1 lists several characteristics of storage encryption technologies as a means for comparing the types of technologies described in this publication.

[26] When multiple encryption methods are used simultaneously, the cryptographic keys used by the encryption methods are usually different.

Table 3-1. Characteristics of Storage Encryption Technologies

Characteristic	Full Disk Encryption	Volume Encryption	Virtual Disk Encryption	File/Folder Encryption
Typical platforms supported	Desktop and laptop computers	Desktop and laptop computers, volume-based removable media (e.g., USB flash drives)	All types of end user devices	All types of end user devices
Data protected by encryption	All data on the media (data files, system files, residual data, and metadata)	All data in the volume (data files, system files, residual data, and metadata)	All data in the container (data files, residual data and metadata, but not system files)	Individual files/folders (data files only)
Mitigates threats involving loss or theft of devices?	Yes	Yes	Yes	Yes
Mitigates OS and application layer threats (such as malware and insider threats)?	No	If the data volume is being protected, it sometimes mitigates such threats.* If the data volume is not being protected, then there is no mitigation of these threats.	It sometimes mitigates such threats*	It sometimes mitigates such threats*
Potential impact to devices in case of solution failure	Loss of all data and device functionality	Loss of all data in volume; can cause loss of device functionality, depending on which volume is being protected	Loss of all data in container	Loss of all protected files/folders
Portability of encrypted information	Not portable	Not portable	Portable	Often portable

** These storage encryption technologies can only protect the files against some OS and application layer threats if the user has not been authenticated in this session to access the files. If a single sign-on solution is used, then generally the user is authenticated to the storage encryption technology during OS login, so the files are not protected against these threats once OS login occurs. If a separate authentication solution is used, the files are protected until that separate authentication is performed.*

When selecting storage encryption technologies, an organization should take into consideration the extent to which each technology will require the infrastructure and end user devices to be changed. For example, using some technologies requires deploying additional servers and installing software on the devices to be protected, while other technologies can use existing servers, as well as software built into the devices to be protected, such as FIPS approved encryption features built into the devices' operating systems. Generally, the more extensive the changes are to the infrastructure and devices, the more likely it is that the storage encryption technology will cause a loss of functionality or other problems with the devices. When evaluating solutions, organizations should compare the loss of functionality with the gain in security capabilities and decide if the tradeoff is acceptable. Technologies that require extensive changes to the infrastructure and end user devices should generally be used only when other technologies cannot meet the organization's needs.

The following are use cases that highlight the types of storage encryption technologies that may be suitable for certain situations. Each use case presents a brief scenario, including the threats that need to be mitigated, and then proposes possible solutions, both storage encryption technologies and alternate solutions (if applicable). Each use case only lists high-level solutions that may be feasible, and is not

intended to imply that other solutions are not possible or that these solutions are preferable to others. Each use case also omits security controls that are universal to the solutions, such as user awareness and general endpoint security (e.g., patching, antivirus software, access control, physical security of endpoint devices), as well as key management (for example, generating keys and securely deploying them to devices).

3.3.1 Use Case 1: Sharing a Laptop

Three users share a laptop. One of the users uses the laptop to access data that the other two users are not authorized to access. For this data, the major threats that the organization needs to mitigate are an insider threat from the other two users, and unauthorized disclosure of data from the loss or theft of the laptop. Possible solutions include the following:

- Implement volume, virtual disk, or file/folder encryption on the laptop. Protect the first user's data using the storage encryption software, with the authentication and cryptographic keys implemented so that only the first user, and not the other two users, can access the protected data. If there is concern about the first user always remembering where to save files, configure the laptop's access control so that the first user's data is all saved to a particular location, and protect that location with the storage encryption software.

- Store the data on external media, such as a flash drive or external hard drive, and use volume, virtual disk, or file/folder encryption to protect the media. The user needs to protect physical access to the media and to remember to save new or modified data to the media.

- Store the data on a remote system and give the first user access to the data through secured means (e.g., VPN). Provide the data in such a way that it is not saved to the laptop (e.g., the user views and modifies the remote data through a Web interface).

3.3.2 Use Case 2: Transferring Files Between Computers

A user edits documents using both a desktop PC at the organization's office and a personally owned computer at home. The user transfers documents between the computers on a daily basis using a USB flash drive. The two computers run different types of OSs. For the documents, the major threat that the organization needs to mitigate is unauthorized disclosure of data from loss or theft of the user's flash drive. Possible solutions include the following:

- Acquire and use a flash drive with self-contained storage encryption capabilities, such as encryption software and secure key storage.

- Acquire a volume, virtual disk, or file/folder encryption solution that will work on both PCs, and deploy it. Encrypt the documents using the solution and store the encrypted data on a flash drive.

3.3.3 Use Case 3: Sharing Data with Contractor

A user wants to provide a contractor with copies of large data sets on a daily basis because the contractor has no direct access to the system containing the data. The user will copy the data onto removable media for the contractor.[27] For this data, the major threat that the organization needs to mitigate is unauthorized disclosure of data from loss or theft of the removable media. Possible solutions include the following:

[27] Another use case, with similar possible solutions, is a user that needs to protect backups of a PC from loss or theft.

- Deploy virtual disk or file/folder encryption software to the user and contractor's computers. Encrypt the data using the software and burn the encrypted data onto CDs or DVDs.
- Acquire USB flash drives or external hard drives that have built-in storage encryption capabilities. Store the copies of the data on the encrypted drives.
- Acquire USB flash drives or external hard drives. Deploy virtual disk, volume, or file/folder encryption software to the user and contractor's computers. Encrypt the data using the software and store it on the drives.

3.3.4 Use Case 4: Traveling with a Laptop

A user occasionally travels on behalf of the organization and carries a laptop that contains sensitive data. For this data, the major threat that the organization needs to mitigate is unauthorized disclosure of data from the loss or theft of the laptop. Possible solutions include the following:

- Use the laptop's OS access control features to strictly limit where the user can save files. Implement volume, virtual disk, or file/folder encryption on the laptop to protect the user's files.
- Implement FDE on the laptop, and require pre-boot authentication.
- Provide the user with a loaner laptop when needed for travel. Protect the user's sensitive data on the laptop using either of the methods described above. When the user returns from travel, wipe and rebuild the loaner laptop to remove any traces of sensitive data from it. Using a loaner laptop in this way is particularly helpful if the laptop is being used in hostile environments, where the laptop is at greater risk of being compromised.

3.3.5 Use Case 5: Traveling with a Dual-Boot Laptop

A user frequently travels on behalf of the organization and carries a laptop that contains sensitive data. The laptop is dual-boot, using two OSs. For the user's data, the major threat that the organization needs to mitigate is unauthorized disclosure of data from the loss or theft of the laptop. Possible solutions include the following:

- Use the OS access control features of each OS to strictly limit where the user can save files. Implement volume, virtual disk, or file/folder encryption on both of the laptop's OSs to protect the user's files.
- Implement an FDE solution that supports dual-boot configurations.
- Convert the laptop to be single-boot, and access the second (removed) OS through a virtual machine run by the primary OS. See use case 4 for further information on protecting the laptop.

3.4 Storage Encryption Technology Management

Most storage encryption deployments are managed centrally. Centralized management is most often performed through special management utilities provided by the storage encryption vendor. If the storage encryption solution is built into the devices' OSs, then it could be managed through the mechanisms already in place to manage OS configurations. The capabilities of centralized management utilities for storage encryption technologies vary considerably. Examples of commonly implemented capabilities are as follows:

- Deploying storage encryption software to additional devices
- Updating storage encryption software (e.g., patching, upgrading)
- Configuring storage encryption software, such as specifying encryption algorithms and setting authentication policies (in some cases, the policies are specific for types of devices, groups of users, and/or individual users)
- Managing storage encryption authenticators and cryptographic keys[28]
- Collecting and reviewing storage encryption-related logs
- Recovering stored information from device failures
- Performing routine system maintenance
- Enabling the encryption of data and managing encrypted storage
  - For FDE, this involves encrypting the computer's hard drive. Some products allow the initial encryption to be done while the computer is in use, but it can cause a substantial performance impact if not configured properly, and additional hard drive space may be needed. Other products require that the computer not be in use while the drive is initially encrypted.
  - For volume encryption, this could involve either encrypting an existing volume, or creating a new volume, encrypting it, and then having the user add files to the volume as needed.
  - For virtual disk encryption, this simply involves creating a container. Files can then be added to the container by the user as needed.
  - For file/folder encryption, this could involve encrypting existing files, setting up an encrypted folder for future files, or establishing policies to automatically encrypt certain types of files.

Some storage encryption products, particularly ones intended for standalone deployment, can be managed locally. Local management is typically performed by a system administrator (for managed devices) or a user (for unmanaged devices) who has physical access to the device running the storage encryption technology. A common local management task is recovering data; many products allow users to recover their own data by running a recovery utility. For example, a device using FDE might experience a failure that prevents it from booting the OS; a user could run a recovery utility from the pre-boot environment or a CD to extract the data from the device.

Organizations may choose to deploy storage encryption without a centralized management capability and perform all management locally. This is generally acceptable for standalone deployments and very small-scale deployments, particularly ones that need to be done quickly, without waiting for a centralized management infrastructure to be implemented. However, for all other deployments, centralized management is recommended because it is more effective and efficient for most management tasks, including policy verification and enforcement, key management, authenticator management, and data recovery.

[28] Section 4 contains additional information on cryptography, key management, and authentication.

4. Storage Encryption Technology Planning and Implementation

This section discusses considerations for planning and implementing storage encryption technologies for end user devices. As with any new technology deployment, storage encryption technology planning and implementation should be addressed in a phased approach. A successful deployment can be achieved by following a clear, step-by-step planning and implementation process. The use of a phased approach for deployment can minimize unforeseen issues and identify potential pitfalls early in the process. This model also allows for incorporating advances in new technology and adapting the technology to the ever-changing enterprise. The following is an example of planning and implementation phases:

1. **Identify Needs.** The first phase involves identifying the needs to encrypt storage on end user devices, determining which devices and data need protection, and identifying related requirements (e.g., minimum performance). This phase also involves determining how that need can best be met (e.g., FDE, virtual disk encryption) and deciding where and how the security should be implemented.

2. **Design the Solution.** The second phase involves all facets of designing the solution. Examples include architectural considerations, authentication methods, cryptography policy, and supporting security controls.

3. **Implement and Test a Prototype.** The next phase involves implementing and testing a prototype of the designed solution in a lab or test environment. The primary goals of the testing are to evaluate the functionality, performance, scalability, and security of the solution, and to identify any issues with the components, such as interoperability issues.

4. **Deploy the Solution.** Once the testing is completed and all issues are resolved, the next phase includes the gradual deployment of the storage encryption technology throughout the enterprise.

5. **Manage the Solution.** After the solution has been deployed, it is managed throughout its lifecycle. Management includes maintenance of the storage encryption components and support for operational issues. The lifecycle process is repeated when enhancements or significant changes need to be incorporated into the solution.

This document does not describe the planning and implementation process in depth because the same basic steps are performed for any security technology. For example, the document assumes that the organization has already determined what information needs to be protected and assigned an impact level from FIPS 199 to the information. This section only highlights those considerations that are particular to storage encryption for end user devices.

4.1 Identify Needs

The purpose of this phase is to identify the needs to protect information stored on end user devices and determine how those needs can best be met. Requirements specific to storage encryption that should be considered include the following:

- **External Requirements.** The organization may be subject to oversight or review by another organization that requires storage encryption. An example is a legal requirement to protect stored PII.

- **System and Network Environments.** It is important to understand the characteristics of the organization's system and network environments so that storage encryption solutions can be selected that will be compatible with them and able to provide the necessary protection. Aspects to consider include the following:

- The characteristics of the devices that need protection, especially the OSs, applications, and filesystems they use, and their hardware capabilities and characteristics

- The technical attributes of the interfaces of other systems with which the storage encryption solution might be integrated, such as authentication services, centralized logging servers and security information and event management (SIEM) software, and patch management software

- **Support Limitations.** The organization should identify any negative impacts that storage encryption technologies could have on existing vendor support mechanisms. For example, installing a storage encryption technology onto an end user device could violate the terms of a support contract for existing software on the end user device or void a warranty for another product used on or with the end user device.

The outcome of the organization's analysis should be a determination of which files or types of files need to be encrypted and which types of threats the storage encryption software should protect against, stating the concerns as specifically as possible. For example, the organization may decide to encrypt sensitive information on all devices used outside the organization's facilities and to encrypt certain types of sensitive information on devices used from any location.

The analysis should also lead to the identification of the type or types of storage encryption technologies that can meet the organization's security needs. Another outcome of the analysis is the documentation of the requirements for the storage encryption technologies themselves, including security capabilities (e.g., authentication, cryptography, key management), performance requirements, management requirements (including reliability, interoperability, scalability), the security of the technology itself, usability (by both administrators and users), and maintenance requirements (such as applying updates).

In most cases, a single storage encryption product cannot meet all of the organization's identified needs. For example, the organization may need to protect information on devices running several different OSs, yet no appropriate product can work on all those platforms. Some devices might also not meet the minimum hardware requirements for storage encryption products. Organizations can solve this problem in several ways, such as acquiring multiple products, using multiple types of storage encryption technologies, replacing older devices, or identifying compensating controls to be used instead of storage encryption that provide the same level of protection. Appendix A discusses some potential alternatives to storage encryption. Organizations should ensure that effective solutions are identified for all the types of end user devices that need their stored information protected, if possible, and that a waiver process is created for unusual cases that cannot be addressed by the identified solutions.

4.2 Design the Solution

Once the needs have been identified and the appropriate type(s) of storage encryption technology have been chosen, the next phase is to design a solution that meets the needs. If these design decisions are incorrect, then the storage encryption implementation will be more susceptible to compromise. Major aspects of solution design that are particularly important for storage encryption are as follows:

- **Cryptography.** Encryption and integrity protection algorithms must be selected, as well as the key strength for algorithms that support multiple key lengths. Key management and protection is another important component of solution design. Section 4.2.1 contains additional cryptography information.

- **Authentication.** Authentication methods must be chosen for users and administrators. Decisions also need to be made regarding the protection of the authenticators themselves. Section 4.2.2 contains a more detailed discussion of authentication.

- **Solution architecture.** The architecture of the storage encryption implementation refers to the selection of devices and software to provide storage encryption services and the placement of centralized elements within the existing network infrastructure, such as authentication credential servers, Web servers for self-service recovery, and management servers. Each end user device must have hardware and/or software that provides protection for the stored information. Designing the architecture includes component placement, redundancy, reliability, and interoperability.

- **Other security controls.** These support and complement the storage encryption implementation. For example, organizations should have policies regarding acceptable usage of storage encryption technologies. Organizations may also set minimum security standards for end user devices, such as mandatory host hardening measures and patch levels, and specify security controls that must be employed, such as host-based personal firewalls, antivirus software, and antispyware software.

- **Minimum requirements for end user devices.** The minimum requirements for the hardware, OS, and supporting software should be defined. They should be based on the requirements supplied by the product vendor and the organization's performance requirements.

Another aspect of solution design is planning the logistics of the solution's deployment. For example, the organization may need to replace devices that do not meet minimum requirements or run on a platform that the organization will not support for storage encryption. This could cause out-of-cycle upgrades or replacements of hardware, OSs, and supporting software. Another logistical consideration is how the solution will be deployed to end user devices. If devices need to be updated locally, such as upgrading the OS, replacing the hard drive, backing up user files, or installing storage encryption software, then organizations need to plan who will perform these actions and when and where the work will be done. Some organizations may need to set up staging areas and get additional personnel to perform this work.

4.2.1 Cryptography

Storage encryption technologies use one or more cryptographic keys to encrypt and decrypt the data that they protect. The number of keys and the types of keys used are product and implementation-dependent. For example, public key cryptography uses a pair of keys, and symmetric cryptography uses a single key. Some products support the use of a recovery key that can be used to recover the encrypted data if the regular key is lost. Also, some technologies permit encrypted storage to be shared by multiple users, which could be enabled by having a different key for each user. Often, users' keys are not directly used to decrypt their stored data; instead, those keys are used to decrypt another key, which in turn is used to decrypt the stored data.

If a key is lost or damaged, it may not be possible to recover the encrypted data. Therefore, organizations need to ensure that all keys used in a storage encryption solution are secured and managed properly to support the security of the solution. Organizations should perform extensive planning of key management processes, procedures, and technologies before implementing storage encryption technologies. This planning should include all aspects of key management, including key generation, use, storage, recovery, and destruction.[29] Organizations should carefully consider how key management practices can support the recovery of encrypted data if a key is inadvertently destroyed or otherwise becomes unavailable (such as a user unexpectedly leaving an organization or losing a cryptographic token containing a key). An example of recovery preparation is storing duplicates of keys in a centralized, secured key repository or on physically secured removable media. Organizations planning on encrypting

[29] NIST SP 800-57, *Recommendation for Key Management*, provides detailed information on key management planning, algorithm selection and appropriate key sizes, cryptographic policy, and cryptographic module selection. Organizations may be able to use the same or similar key management processes for end user devices' storage encryption, virtual private network (VPN) clients, and wireless client configuration.

removable media also need to consider how changing keys will affect access to encrypted storage on the media and develop feasible solutions, such as retaining the previous keys in case they are needed.

Another decision that may need to be made is where the local keys should be stored. For some encryption technologies, such as FDE and many file/folder encryption products, there are often several options for key location, including the local hard drive, a USB flash drive, a cryptographic token, or a Trusted Platform Module (TPM) chip.[30] Some products also permit keys to be stored on a centralized server and retrieved automatically after the user authenticates successfully. For volume and virtual disk encryption, the main encryption key is often stored encrypted within the volume or container itself. Some storage encryption products do not store a key; instead, they perform a cryptographic hash function on the password entered by the user and use that hash as the key.

Organizations need to ensure that access to keys (other than those intended to be available to others, such as public keys) is properly restricted. Storage encryption solutions should require the use of one or more authentication mechanisms, such as passwords, smart cards, and cryptographic tokens, to decrypt or otherwise gain access to a storage encryption key. The keys themselves should be logically secured (e.g., encrypted) or physically secured (e.g., stored in a tamper-resistant cryptographic token). The authenticators used to retrieve keys should also be secured properly.

In addition to key management, there are several other aspects of cryptography that need to be considered when planning a storage encryption solution. Setting the cryptography policy involves choosing encryption and integrity protection algorithms and key lengths.[31] Federal agencies must use FIPS-approved algorithms contained in validated cryptographic modules.[32] Whenever possible, AES[33] should be used for the encryption algorithm because of its strength and speed. Several FIPS-approved algorithms are available for integrity checking, including HMAC-SHA, Cipher-Based Message Authentication Code (CMAC), and Counter with Cipher Block Chaining-Message Authentication Code (CCM).[34] Organizations should consider how easily the solution can be updated when stronger algorithms and key sizes become available in the future.

4.2.2 Authentication

There are two types of authentication important to storage encryption. Administrators authenticate so that they can perform storage encryption management functions, including reconfiguring and updating encryption software, managing user accounts, and recovering encrypted data.[35] Users authenticate so that they can access encrypted information. If a single authenticator is used (often a user ID and password, sometimes a token), that authenticator typically grants the storage encryption software access to the key used to encrypt and decrypt the stored information. If two-factor authentication is used, typically one of

---

30 A *TPM chip* is a tamper-resistant integrated circuit built into some motherboards that can perform cryptographic operations (including key generation) and protect small amounts of sensitive information, such as passwords and cryptographic keys. As of this writing, no FIPS-approved TPM chips are yet available.

31 NIST SP 800-21, Second Edition, *Guideline for Implementing Cryptography in the Federal Government*, presents guidelines for selecting, specifying, employing, and evaluating cryptographic protection mechanisms in Federal information systems. It defines a process for selecting cryptographic products and discusses implementation issues, including solution management, key management, and authentication. NIST SP 800-21 is available at http://csrc.nist.gov/publications/nistpubs/.

32 The Cryptographic Module Validation Program (CMVP) at NIST coordinates FIPS 140-2 testing; the CMVP Web site is located at http://csrc.nist.gov/cryptval/. See http://csrc.nist.gov/cryptval/des.htm for information on FIPS-approved symmetric key algorithms, and http://csrc.nist.gov/cryptval/dss.htm for information on digital signature algorithms. FIPS 140-2, *Security Requirements for Cryptographic Modules*, is available at http://csrc.nist.gov/publications/fips/.

33 For more information, read FIPS 197, *Advanced Encryption Standard (AES)*, at http://csrc.nist.gov/publications/fips/.

34 Additional information on these algorithms is available at http://csrc.nist.gov/CryptoToolkit/modes/.

35 The term "administrators" is used generically to refer to the individuals responsible for managing the storage encryption solution. Some organizations use more specific terms for these individuals, such as "cryptographic officers".

the factors grants access to information secured in another factor, which is then used to gain access to the storage encryption key. For example, a PIN or password could be used to retrieve a key from a smart card or cryptographic token; that key could then be used to decrypt the storage encryption key.[36]

Some storage encryption products allow the use of multiple user IDs on a single device. If the IDs are tied to a single storage encryption key, then each user can access the same protected information. If each ID is linked to a separate key, then the access is dependent on how the keys are used—for example, a container could be encrypted using a single key so that only one user can access it, or encrypted using several keys so that users can share the contents of the container.

For storage encryption authentication, organizations often want to leverage existing enterprise authentication solutions (e.g., Active Directory, RADIUS, PKI, Personal Identity Verification [PIV] cards) instead of adding another authenticator for users. Generally, using an existing authentication solution is acceptable only if it provides multi-factor authentication. Using a single-factor authenticator for multiple purposes significantly weakens the protection that authentication provides.[37] For example, reusing a user's OS password for pre-boot authentication in an FDE deployment would allow an attacker to learn only a single password to gain full access to the device's information. The password could potentially be acquired through technical methods, such as infecting the device with malware, or through physical means, such as watching a user type in a password in a public location. Another example is having a volume, virtual disk, or file/folder encryption product use the OS's authentication for single sign-on (SSO). Once a user authenticates to the OS at login, the user can access the encrypted files without further authentication, so the security of the solution is heavily dependent on the strength of the OS authenticator. Organizations should carefully consider the security implications of using the same single-factor authenticator for multiple purposes. In particular, organizations should not use email passwords and other passwords sometimes transmitted in plaintext as single-factor authenticators for storage encryption. Also, organizations concerned about targeted attacks, such as someone stealing a particular laptop to gain access to a specific user's data, should not use only passwords for storage encryption authentication because of the relative ease of capturing a user's password (e.g., watching a password being typed).

Organizations also need to ensure that the storage encryption authenticators are protected properly. This includes both technical mechanisms, such as encrypting passwords or storing cryptographic hashes of passwords, and operational and management mechanisms. For example, policy should state and users should be made aware that authenticators should not be stored in proximity of the end user device (e.g., a password should not be on a piece of paper in a laptop case), and that for two-factor authentication, multiple authenticators should not be stored with each other (e.g., a password or PIN should not be written on the back of a hardware token).

Because authentication controls access to storage encryption keys, the loss of authenticators can prevent access to the encrypted data. Organizations should determine how the loss of authenticators (both user and administrator-level) will be handled before implementing storage encryption. Most products offer recovery mechanisms for password-based user authentication. For example, a user that has forgotten a password chooses a recovery option on the protected computer. The computer provides a special code to the user, who then uses another computer to access the organization's storage encryption recovery Web

---

[36] The following authentication methods for storage encryption are listed from weakest to strongest based on their general effectiveness: single sign-on, unique password or PIN, token, token with unique password or PIN. See NIST SP 800-63, *Electronic Authentication Guideline*, for additional information (http://csrc.nist.gov/publications/nistpubs/).

[37] In many cases, single-factor authentication will not be an option because the sensitivity of the data being protected will necessitate the use of multi-factor authentication. Using the same single-factor authenticator for storage encryption and other purposes may be acceptable if appropriate compensating controls are used, such as the device using storage encryption residing only in physically secured office space.

site. The user provides the code to the Web site and gives proof of identity, such as answering questions about personal preferences (e.g., favorite color) that the user has previously configured. The Web site then provides the user's password or a one-time recovery code that the user enters into the computer to regain access. A similar process can also be performed by having users call a help desk instead of accessing a particular Web site.

For user authentication methods other than password-based, recovery is often more difficult, especially if the user is not at the organization's facilities. Some storage encryption products allow the password-based authentication recovery mechanisms to be used and permit the user to temporarily use password-based authentication. However, because this is generally a reduction in the strength of authentication, many organizations do not permit its use. This means that a loss of authenticator could cause an extended loss of availability to the data, until the user can receive a new authenticator (e.g., smart card, cryptographic token) and an administrator can configure the device to use the new authenticator.

Recovery mechanisms increase the availability of the storage encryption solution for individual users, but they can also increase the likelihood that an attacker can gain unauthorized access to encrypted storage by abusing the recovery mechanisms. Organizations should consider the tradeoff between availability and security when selecting and planning recovery mechanisms.

Some storage encryption products also offer protection against authentication-guessing attempts. For example, if there are too many consecutive failed authentication attempts, some products can either lock the computer for a period of time or increase the delay between attempts. In particularly high-security situations, some products can be configured so that too many failed attempts causes the product to wipe all the protected data from the device. This approach strongly favors security over functionality.

4.3 Implement and Test Prototype

After the solution has been designed, the next step is to implement and test a prototype of the design. Ideally, implementation and testing should first be performed on lab or test devices. Only implementations in final testing should be conducted on production devices. Aspects of the solution to evaluate include the following:

- **Protection.** Each type of information that needs protection should be protected in accordance with the information gathered during the Identify Needs phase. This should be verified by using forensic tools to confirm that the information is encrypted. For devices that use FDE and offer hibernation, standby, or other "suspend" modes, encryption should be verified in each mode; if the mode does not write the contents of memory out to disk and encrypt it, then the information may be readily available unencrypted.[38]

- **Authentication.** Performing robust testing of authentication is important, especially for more complex authentication solutions that depend on centralized authentication services; a loss of those services could cause a loss of storage encryption services as well.

- **OS and Application Compatibility.** The solution should not break or interfere with the use of existing OS configurations and software applications. Examples of applications that may be particularly problematic are, for FDE, disk-level software tools, asset management software, and dual-boot configurations, and for all storage encryption technologies, backup utilities, forensic tools, and other storage encryption programs.

[38] This can be addressed by configuring the device not to use modes that maintain the data in an unencrypted format.

- **Management.** Administrators should be able to configure and manage all components of the solution effectively and securely. It is particularly important to evaluate the ease of deployment and configuration, including how easily the solution can be managed as the solution is scaled to larger deployments. Another concern is the ability of administrators to disable configuration options so that users cannot circumvent the intended security. Management concerns should include the effects of patching/upgrading software, changing software settings (e.g., changing cryptographic algorithms or key sizes), uninstalling or disabling encryption software, changing encryption/decryption keys, and changing user or administrator passwords.

- **Logging.** The logging and data management functions should function properly in accordance with the organization's policies and strategies.

- **Performance.** The solution should be able to provide adequate performance during normal and peak usage. Testing should incorporate a variety of devices, OSs, and applications, especially those that are most likely to be affected by performance issues, such as those that manipulate large files.

- **Security of the Implementation.** The storage encryption implementation itself may contain vulnerabilities and weaknesses that attackers could exploit. Organizations with high security needs may want to perform extensive vulnerability assessments against the storage encryption components. Another common security concern is the security of the authenticators and cryptographic keys.

- **Recovery.** The solution should be tested to determine how well it can recover from failures, such as lost or forgotten authenticators, lost keys, device hardware or software failure/damage, and power loss.

- **Interoperability.** For a solution that will protect removable media that will be used on multiple devices, the organization should ensure that information encrypted on the media by one device can be decrypted by another device after authenticating successfully.

- **Operational Impacts.** Organizations should determine how the solution might impact operations, such as impeding technical support and incident response actions involving end user devices.

Organizations should consider implementing the components in a test environment first, instead of a production environment, to reduce the likelihood of implementation problems disrupting the production environment. When the components are being deployed into production, organizations should initially use encryption on a small number of hosts. Deploying it to many hosts at once might overwhelm the management servers or identify other bottlenecks through loss of availability. Many of the problems that occur are likely to occur on multiple hosts, so it is helpful to identify such problems either during the testing process or when deploying the first hosts, so that those problems can be addressed before widespread deployment. A phased deployment is also helpful in identifying potential problems with scalability.

Actions that may be prudent to perform before installing storage encryption software on end user devices include the following:

- Ensure that any files to be encrypted can be restored. Examples include backing up user files and having a disk image for the computer's OS.

- Replace hardware components (e.g., replace an old hard drive) or the whole device if necessary (e.g., equipment that is considered too slow or unreliable).

- Ensure that the OS is secured properly, including that it is fully patched and that other necessary security controls, such as antivirus software, are installed and configured properly. If the OS is not secured properly, the device is more likely to be compromised, which could weaken the protection provided by the storage encryption solution.

- Scan the device for malware and either remove any malware that is detected or rebuild the device.

4.4 Deploy the Solution

Once testing is complete and any issues have been resolved, the next phase of the planning and implementation model involves deploying the solution. A prudent strategy is to gradually migrate devices and users to the new solution. The phased deployment provides administrators an opportunity to evaluate the impact of the solution and resolve issues prior to enterprise-wide deployment. It also provides time for the IT staff (e.g., system administrators, help desk) and users to be trained.

Most of the issues that can occur during deployment are the same types of issues that occur during any large IT deployment. In addition to potential problems described earlier in this publication, another typical issue is that storage encryption technologies might not work properly on some devices because of incompatibilities with particular hardware configurations.

4.5 Manage the Solution

The last phase of the planning and implementation model is the longest lasting. Managing the solution involves operating the deployed solution and maintaining the security storage architecture, policies, software, and other solution components. Examples of typical actions are as follows:

- Testing and applying patches to storage encryption software. It is beneficial to have at least one host (one for each type of platform) that is used strictly for testing updates. This can help to identify possible conflicts between an update and the normal functions of devices. Software updates should be tested and deployed using the same practices that would be used for updating any other major security controls, such as antivirus software.

- Deploying storage encryption technologies to additional types of devices

- Configuring additional devices to use the technologies

- Performing key management duties (e.g., issuing new credentials, revoking credentials for compromised systems or departing users)

- Performing recovery actions (e.g., regaining access to encrypted data when the authenticator has been lost or the storage media has been damaged)

- Adapting the policies as requirements change. An example is switching to a stronger encryption algorithm or increasing the key size.

- Monitoring the storage encryption components for operational and security issues

- Periodically performing testing to verify that storage encryption is functioning properly

- Performing regular vulnerability assessments

- Receiving notifications from vendors of security problems with storage encryption components, and responding appropriately to those notifications

- Preparing devices for retirement or disposal. Devices and media that use storage encryption technologies should be sanitized or destroyed, even for devices using FDE. As mentioned in Section 3.2, relying on storage encryption to protect data without regularly maintaining the encryption solution is not recommended. Another problem with relying on storage encryption to protect data on retired or disposed devices is that all copies of all keys used for storage encryption would need to be destroyed, which may be very difficult.

User files on the device should be backed up before major maintenance actions are performed, such as installing or upgrading storage encryption software and changing encryption algorithms or key sizes.

Appendix A—Alternatives to Encrypting Storage on End User Devices

Section 3 describes commonly used technologies for storage encryption—full disk, virtual disk and volume, and file/folder encryption. Organizations should not feel compelled to use only these methods to encrypt stored information; there are many other acceptable methods. The following are some examples:

- Applications can encrypt the information that they store. For example, a commercial off-the-shelf (COTS) backup utility might be capable of encrypting its backups, and a compression utility might have an option to encrypt archives that a user creates. Another example is a database that can be configured to encrypt fields that contain sensitive information. An application could also store sensitive information in an alternate format, such as cryptographic hashes of passwords instead of the passwords themselves.

- Sensitive information could be accessed only through a virtual machine and stored as part of the virtual machine. If the virtual machine software itself does not provide an encryption capability, the virtual machine data, which is a single file, could be protected through storage encryption software.

In some cases, organizations may decide that the best way to address the problem of protecting sensitive information on end user devices is not to store the information on the devices. Examples of how this might be implemented include the following:

- Preventing access to sensitive information from higher-risk devices, such as mobile devices

- Using a thin client solution, such as terminal services, a thin Web-based application, or a portal, to access the information, and configuring the thin client solution to prohibit file transfers of the sensitive information to the end user device

- Configuring the organization's devices (including desktop computers) to prevent writing sensitive information to removable media, such as CDs or USB flash drives, unless the information is properly encrypted

- Permitting users to access files or databases containing sensitive information only through well-secured applications that restrict access as tightly as possible. For example, suppose that an organization has a database containing thousands of records on employees' benefits. Instead of allowing a user to have full and direct access to the database, which could allow the user to transfer all the database records to the user's device, the organization could permit the user to access only the necessary records and record fields. If the user only needs access to general demographic information, and does not need to access any information related to the employees' identities, then the user would not be able to access any sensitive information.

- Removing unneeded sensitive information from files or databases.

Organizations should be aware that if sensitive information can be viewed from end user devices, the information is still at some risk of exposure, even if it is not stored there. The information could be recorded in screen captures, printed, or monitored by malware, for example.

Organizations should also be aware that the use of general access control mechanisms is typically insufficient to protect sensitive information on end user devices. For example, a password generally cannot be used instead of cryptography to protect stored information. Although requiring a BIOS password can prevent an attacker from booting a computer regularly, the attacker could still access the information by placing the storage media in a different computer. Using an OS password is also

ineffective because forensic tools can examine the storage media directly. OS controls such as access control lists may deter users from accessing each other's files, but they can also be circumvented by using forensic tools. Properly designed and implemented encrypted storage cannot be decrypted by forensic tools.

Appendix B—Glossary

Selected terms used in the publication are defined below.

Container: The file used by a virtual disk encryption technology to encompass and protect other files.

End User Device: A personal computer (desktop or laptop), consumer device (e.g., personal digital assistant [PDA], smart phone), or removable storage media (e.g., USB flash drive, memory card, external hard drive, writeable CD or DVD) that can store information.

File: A collection of information logically grouped into a single entity and referenced by a unique name, such as a filename.

File Encryption: The process of encrypting individual files on a storage medium and permitting access to the encrypted data only after proper authentication is provided.

Filesystem: A mechanism for naming, storing, organizing, and accessing files on logical volumes.

Folder: An organizational structure used by a filesystem to group files.

Folder Encryption: The process of encrypting individual folders on a storage medium and permitting access to the encrypted files within the folders only after proper authentication is provided.

Full Disk Encryption (FDE): The process of encrypting all the data on the hard drive used to boot a computer, including the computer's operating system, and permitting access to the data only after successful authentication with the full disk encryption product.

Malware: A program that is inserted into a system, usually covertly, with the intent of compromising the confidentiality, integrity, or availability of the victim's data, applications, or operating system.

Master Boot Record (MBR): A special region on bootable media that determines which software (e.g., operating system, utility) will be run when the computer boots from the media.

Metadata: Data about data; in the context of a filesystem, information regarding files and folders themselves, such as file and folder names, creation dates and times, and sizes.

Partitioning: The act of logically dividing a media into portions that function as separate units.

Pre-Boot Authentication (PBA): The process of requiring a user to authenticate successfully before decrypting and booting an operating system.

Residual Data: Data from deleted files or earlier versions of existing files.

Sector: The smallest unit that can be accessed on media.

Storage Security: The process of allowing only authorized parties to access stored information.

Trusted Platform Module (TPM) Chip: A tamper-resistant integrated circuit built into some computer motherboards that can perform cryptographic operations (including key generation) and protect small amounts of sensitive information, such as passwords and cryptographic keys.

Virtual Disk Encryption: The process of encrypting a container, which can hold many files and folders, and permitting access to the data within the container only after proper authentication is provided.

Volume: A logical unit of storage comprising a filesystem.

Volume Encryption: The process of encrypting an entire volume and permitting access to the data on the volume only after proper authentication is provided.

Whole Disk Encryption: See "Full Disk Encryption".

Appendix C—Acronyms

Selected acronyms used in the publication are defined below.

AES	Advanced Encryption Standard
BIOS	Basic Input/Output System
CAVP	Cryptographic Algorithm Validation Program
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CMAC	Cipher-Based Message Authentication Code
CMVP	Cryptographic Module Validation Program
COTS	Commercial Off-the-Shelf
DRM	Digital Rights Management
EFS	Encrypting File System
EPHI	Electronic Protected Health Information
FDE	Full Disk Encryption
FIPS	Federal Information Processing Standards
FISMA	Federal Information Security Management Act
GLBA	Gramm-Leach-Bliley Act
HIPAA	Health Insurance Portability and Accountability Act
HMAC	Keyed-Hash Message Authentication Code
IT	Information Technology
ITL	Information Technology Laboratory
KB	Kilobyte
MBR	Master Boot Record
NIST	National Institute of Standards and Technology
NTFS	New Technology File System
NVD	National Vulnerability Database
OMB	Office of Management and Budget
OS	Operating System
PBA	Pre-Boot Authentication
PBE	Pre-Boot Environment
PDA	Personal Digital Assistant
PII	Personally Identifiable Information
PIN	Personal Identification Number
PKI	Public Key Infrastructure
RADIUS	Remote Authentication Dial In User Service

SHA	Secure Hash Algorithm
SIEM	Security Information and Event Management
SP	Special Publication
SSO	Single Sign-On
TPM	Trusted Platform Module
USB	Universal Serial Bus
VPN	Virtual Private Network

Appendix D—Tools and Resources

The lists below provide examples of tools and resources that may be helpful.

Resource Web Sites

Organization	URL
Computer Forensics Tool Testing (CFTT) Project	http://www.cftt.nist.gov/
Cryptographic Algorithm Validation Program (CAVP)	http://csrc.nist.gov/groups/STM/cavp/index.html
Cryptographic Module Validation Program (CMVP)	http://csrc.nist.gov/groups/STM/cmvp/index.html
Full Disk Encryption mailing list and discussion group	http://www.full-disc-encryption.com/
Modes of Operation for Symmetric Key Block Ciphers	http://csrc.nist.gov/CryptoToolkit/modes/

Resource Documents

Document Title	URL
FIPS 140-2, *Security Requirements for Cryptographic Modules*	http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf
FIPS 180-2, *Secure Hash Standard*	http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf
FIPS 197, *Advanced Encryption Standard (AES)*	http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
NIST SP 800-21-1, *Guideline for Implementing Cryptography in the Federal Government*	http://csrc.nist.gov/publications/nistpubs/800-21-1/sp800-21-1_Dec2005.pdf
NIST SP 800-32, *Introduction to Public Key Technology and the Federal PKI Infrastructure*	http://csrc.nist.gov/publications/nistpubs/800-32/sp800-32.pdf
NIST SP 800-38A, *Recommendation for Block Cipher Modes of Operation-Methods and Techniques*	http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf
NIST SP 800-38B, *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*	http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf
NIST SP 800-38C, *Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*	http://csrc.nist.gov/publications/nistpubs/800-38C/SP800-38C.pdf
NIST SP 800-38D (DRAFT), *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) for Confidentiality and Authentication*	http://csrc.nist.gov/publications/drafts.html
NIST SP 800-53 Revision 1, *Recommended Security Controls for Federal Information Systems*	http://csrc.nist.gov/publications/nistpubs/800-53-Rev1/800-53-rev1-final-clean-sz.pdf
NIST SP 800-57, *Recommendation for Key Management—Part 1: General*	http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf
NIST SP 800-57, *Recommendation for Key Management—Part 2: Best Practices for Key Management Operation*	http://csrc.nist.gov/publications/nistpubs/800-57/SP800-57-Part2.pdf
NIST SP 800-63, *Electronic Authentication Guideline*	http://csrc.nist.gov/publications/nistpubs/800-63/SP800-63V1_0_2.pdf
NIST SP 800-67, *Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher*	http://csrc.nist.gov/publications/nistpubs/800-67/SP800-67.pdf