Global Information Assurance Certification Paper

Analysis of Design for Implementing Confidentiality, Integrity, Authentication, and Non-Repudiation Solutions.

GSEC Assignment Version 1.4b

6-09-03

Name: Ebbe Hansen

1. Introduction.

This paper examines how three types of cryptographic algorithms (checksum/hash, encrypt, and decrypt) can be deployed in a variety of combinations to provide basic Confidentiality, Integrity, Authentication, and Non-repudiation (CIAN) security solutions. Benefits and drawbacks for each analyzed CIAN solution examined, when applying none, one or two keys, together with one of the above-mentioned crypto algorithms, are described in paragraph two and summarized in Table 2-1.

The paper also examines several well-known standards, which have been designed to provide one or more CIAN security services. Each standard is first analyzed and categorized in accordance with the provided CIAN solution(s) and the number of applied keys. The strength of the security design associated with a given standard can subsequently be assessed by evaluating all benefits and drawbacks listed in the Table 2-1 entries that are identified by the CIAN/key combinations derived during the analysis of each standard.

2. Types of Cryptographic based Security Solutions.

While Confidentiality security solutions have been in use for many hundreds of years (mostly in military applications) it is not until recently that a variety of CIAN security implementations have been utilized on a regular basis by individual users and organizations primarily when deploying new computerized information systems.

In this paragraph we will examine how various CIAN solutions are implemented using none, one, or two keys together with one of the three types of cryptographic algorithms mentioned above. Table 2-1 shown at the end of this paragraph, with associated legends, provides an overview of all examined CIAN security solutions.

The labels shown inside the curled parenthesis included below identify a benefit or a drawback. For example, {B7} reference benefit number seven and {D2, D4} reference drawback number two and number four.

2.1 No-Key Crypto Solutions.

The cryptographic algorithms analyzed in this sub-paragraph (which do not require a key when applied) are limited to one-way hash functions like MD5 and SHA.1.

a. Confidentiality.

A hash function applied over a secret password will allow the hash of the password to be stored on a computer hard disk without revealing the clear-text password to intruders who may gain access to the contents of the hard-disk (for instance via a backup tape).

{B1} The hashing of passwords appears to be superior to the single-key encryption solution, if it also requires a secret key to be maintained (see 2.2.a).

b. Integrity.

Any size document can be hashed to provide a document digest which typically is smaller than the document itself. {B2} The digest can then be stored or transmitted safely (sometimes separately from the document) and subsequently be used by a third party to verify the integrity of the original document.

A trusted third-party, which calculates a document digest and stores such digest securely, possibly together with the current time obtained from a trusted time-source, is often referred to as a time stamping service. {B3} A document time-stamp record can be used subsequently as a basis to verify the Integrity of a previously time-stamped document.

c. Authentication.

{B2} A hash function applied over a shared secret will allow the hash of the shared secret to be transmitted over public networks without revealing the value of the shared secret.

To avoid replay attacks, the hash function is often calculated over a shared secret, after it is combined with a challenge value, sometimes also referred to as a Salt. [1] {B4} The Salt can be exchanged freely over public networks without jeopardizing the confidentiality of the shared secret.

An independent party can authenticate a specific user by concluding the user in question must have access to a copy of the correct shared secret. Otherwise, the user will not be able to produce the correct response to a specific challenge. {D1} The authenticating party must, however, also trust that the original user has not "leaked" the shared secret to any third party.

{D2} To guarantee a given user can not easily (by accident or intend) "leak" the shared secret, the secret value must be stored permanently inside a trusted hardware token (e.g. a smart card, cell-phone, etc.), which is designed to calculate the required hash value inside the token when supplied with a Salt value.

d. Non-repudiation.

As non-repudiation typically is associated with generation of some form of electronic signature, we must first examine how an electronic signature typically is produced before considering the possibility of using a hash function to generate an electronic signature.

While the legal requirement for an electronic signature simply refers to methods to prove "consent" to a given electronic transaction, [2] a typical strong electronic signature using Dual Public Key (PK) technology (ref Paragraph 2.3.d) requires the signer to apply a unique private key to the contends (or "digest") of a document, thus producing a non-reputable signature value.

Considering the above, it should be possible to produce a legal electronic signature for a document by generating a hash value over a document "digest" after combining the "digest" with the value of a shared secret, which has previously been assigned to the signing individual.

{D3} The biggest drawback relating to such hash-based signature is the fact that each verifier will need a copy of the shared secret to verify the signature. This would make it possible for the verifier to falsify any such type of signature.

{B5} However, this limitation may not be significant when enabling single-authority organizations, which only need to verify their own employee's compliance with company issued policies. {D3} Any independent individual/organization located outside the organization will not be able to verify such compliance unless they are also provided access to the shared secret in question.

2.2 Single-Key Crypto Solutions.

a. Confidentiality.

{B6} Anyone in possession of a single (encryption) key is able to encrypt a clear-text message or decrypt a cipher-text message previously encrypted, using the same key and algorithm. [3] There are a variety of single-key algorithms and key-sizes to choose from. {D4} So it is not just the key value, but also the applied algorithm and key-size that must be exchanged between any two parties before encrypted information can be decrypted.

{D5} The biggest drawback associated with use of single-key (sometimes call symmetric-key) encryption technology is the rapidly growing numbers of keys that must be exchanged as the number of participating parties grows. {D5} Each individual (in a group of n) must maintain a minimum of n-1 different keys in order to exchange confidential information with all other (n-1) individuals/organizations in such group. This comes to a total of n*(n-1) different keys to be maintained (providing each party wish to manage their set of keys for communicating with all other (n-1) entities).

b. Integrity.

Single key encryption is not well suited to guarantee the integrity of an original clear-text message. One might claim loss of integrity for a transmitted cipher-text should be detected if only part of a message is appearing as "expected" after the decryption

process (e.g. readable text garbled, input data rejected, etc). However, if the integrity of the message is compromised before the encryption process or after the decryption, the encryption/decryption operation offers no integrity protection at all. Adding some form of checksum to the clear-text message (could be a hash-value as described above) is clearly a better way to detect loss of integrity.

c. Authentication.

An entity that uses an assigned unique (encryption) key to encrypt a known set of information (e.g. their name combined with a Salt) can subsequently be authenticated by decrypting such message and verifying the resulting clear-text appears as expected (i.e. contains name, Salt, etc.).

{D4} The drawback for such single-key authentication scheme is the same as described above for the single-key confidentiality solution (i.e. key-values must be shared).

d. Non-repudiation.

A given individual can generate a legal electronic signature over a specific message by applying the secret single-key to the message (i.e. by encrypting the message contents).

{D6} A recipient of an encrypted message will be able to verify that such signature has been applied correctly, only if the recipient is also given means to validate the contents or the integrity of the message. For example, the decrypted message must be "readable" by humans/machines, or it must hold some pre-agreed information (e.g. name, time, Salt, etc.). The hash-value of the message may for example also be provided to the verifier by other (out-of-band) means.

The drawbacks for deploying the single-key signature scheme are described in 2.2.a (excessive key-management burden {D5}) and in 2.1.d (no independent third-party verification possible {D3}).

2.3 Dual-Key Crypto Solutions.

a. Confidentiality.

After the Public (dual) Key cryptographic algorithm(s) was invented in the 1970's, {B7} it became possible for multiple individuals to use the same (public) key to encrypt information that can only be decrypted by the intended recipient using the matching (private) key. {D1} However, it is imperative the owner of the Private Key never shares the key with anyone else. [4]

{B8} The maximum number of public keys required for a given individual to send confidential information to all other users in a group of n, is hereby reduced to n-1. This

is a factor n less than the number of keys required when using the single-key (symmetric) encryption technology (see paragraph 2.2.a for further details).

{B9} Another benefit of applying PK technology is the sender and the recipient can trust nobody else, but the recipient, should be able to decrypt the enciphered information.

{D7} One of the drawbacks associated with use of PK algorithms is the relative high level of processing power that is needed to encrypt and decrypt information of substantial size. This limitation forces most PK technology users to combine the use of PK technology with use of single-key encryption technology when encrypting large messages. This “mixed” crypto solution calls for a single random key to be generated (often called a session key), which is subsequently used to encrypt any large message before they can be transmitted or stored (see. 2.2.a). {B10} Only the session key will then need to be encrypted using the public key for each targeted recipient.

b. Integrity.

Public (dual) key technology is not well suited for guaranteeing Integrity before, during, and after encryption/transmission. The limitations described in paragraph 2.2.b (single-key Integrity solutions) apply likewise when using the dual-key technology.

c. Authentication.

{B11} An individual that uses an assigned unique private key to encrypt a known set of information (e.g. signing a challenge Salt) can subsequently be verified to have access to the private key. When decrypting such (signature) message, using the matching public key, the resulting clear-text should reveal the agreed-upon data.

Additional benefits and drawbacks applying to the dual-key authentication scheme are already covered above for the dual-key confidentiality solution (i.e. no sharing of the private key is required, smaller number of keys to be maintained, private key shakk not be shared – {B7,B8,D1}).

d. Non-repudiation.

A given individual can generate an electronic signature for a specific message or document by encrypting the data using the private key assigned to the individual in question.

{B11} Any verifying party can decrypt the signature using the matching public-key. If the decrypted signature reveals the expected clear-text value (typically the hash of the signed document), the signature of the document is considered valid and binding. {B12, D2} The signing individual will not be able to repudiate a given signature since the private key used to generate the signature is accessible only to the signer (no key sharing required).

{B13} When applying an electronic signature to a large document a “digest” of the entire document is calculated first and then the signature is generated over the resulting “digest”. {D7} This is done primarily to reduce the processing power associated with use of public/private key algorithms (see 2.3.a for drawbacks).

An associated PK Use Policy document will state that an individual which signs a hash-value of a document, will be legally bound, as if the signature was generated over the entire document.

Table 2-1 and associated legends provide an overview of benefits and drawbacks associated with the above-described Security Solutions.

Solution	No-Key	Single Key	Dual key
Confidentiality	B1	B6, D4, D5	B7, B8, B9, B10, D1, D7
Integrity	B2, B3	NR	NR
Authentication	B2, B4, D1, D2	D4	B7, B8, B11 D1
Non-Repudiation	B5, D3	D3, D5, D6	B11, B12, B13 D2, D7

Table2-1, Benefit & Drawback Overview.

Legends to table 2-1:

B1: Storage of hashed password requires no encryption key.
B2: Document “digest” can be transmitted over un-trusted network.
B3: A document time-stamp provides good basis for Integrity checking.
B4: Challenge Salt can be transmitted over un-trusted network.
B5: Shared secret security solutions are well integrated into single-authority trust environments.
B6: Single key encryption/decryption offers fast Confidentiality protection.
B7: Dual key cryptography does not require any private key to be shared.
B8: Dual key cryptography reduces the number of keys to be managed.
B9: Dual key cryptography assures only the private key owner can decrypt an encrypted message.
B10: Using a session key together with public key encryption, reduces processing power needs.
B11: Use of dual key cryptography assures only the private key owner can sign a message.
B12: Use of dual key cryptography guarantees the signer of a message cannot repudiate after signing a message.

B13: A signature generated over a message digest is legally binding as a signature generated over the entire message contents.

D1: The shared or private key shall never be leaked to any third party.
D2: A dedicated hardware token is needed to provide strong protection of shared or private key.
D3: Access to the shared secret is needed to verify a single key signature.
D4: Key value, key-size, and algorithm must be exchanged when using single-key encryption (in addition to the enciphered text).
D5: Use of single key cryptography requires a large number of shared keys to be created and maintained.
D6: Additional Integrity information is needed to validate signatures generated via a single (symmetric) key.
D7: Using dual key cryptography algorithms require a high level of processing power.

NR: Not recommended.

3. Examples of Standards Providing basic CIAN Functionality.

In this paragraph we will examine various cryptographic standards designed to provide one or more of the CIAN security services using one or more of the key and algorithm combinations described in paragraph 2. Each standard described is labeled, with a [C_x] for Confidentiality, [I_x] for Integrity, [A_x] for Authentication, and [N_x] for Non-repudiation, to identify the type of embedded security solution that is embedded. The sub-script x indicates the number of keys used, for example [A_1] identifies an Authentication solution that is using a single-key and [I_0] identifies an Integrity solution using no keys.

3.1 No-Key Crypto Solutions.

a. Hash Algorithms.

Hashing algorithms like MD5 (Request For Comment (RFC)-1321 [5]) and Secure Hash Algorithm 1 (SHA1) [6] are often used to verify data integrity of transmitted, received or stored information. In some applications the hash value is applied as a stand-alone protection mechanism, but in most cases the hash functions are combined with use of other single-key and dual-key security solutions.

CIAN Service: The hash algorithms are primarily used to provide Integrity protection [I_0].

b. Authentication Protocols.

Various authentication protocols have been standardized which apply a hash calculation over a shared secret, often combined with a dynamically applied challenge value.

Here is a small list of hash based authentication protocols that will all preserve the Confidentially of the shared secret.

- PPP/CHAP Protocol.

The PPP Challenge Handshake Authentication Protocol (CHAP, RFC-1994) has been defined to provide shared-secret authentication over existing Point-to-Point Protocol (PPP) links. It was originally defined to be used with the MD5 hash algorithm but was later extended for use with various Microsoft applications.

CIAN Service: The CHAP provides Authentication, and Confidentiality of shared secret $[A_0C_0]$.

- Authenticated POP (APOP)

The Post Office Protocol (POP, RFC-1460) is still used by many e-mail clients. Although it includes an option to apply shared-secret authentication, this option has not been supported well by many Internet Service Providers (ISP) or e-mail client vendors. As a result, many e-mail subscribers are today still submitting the logon name and password in the clear, each time they connect to download e-mail from their local (public) ISP.

CIAN Service: The APOP provides Authentication, and Confidentiality of shared secret $[A_0C_0]$.

- HMAC

The Keyed-Hash for Message Authentication (HMAC, RFC-2104) has been developed to provide combined Integrity and Authentication protection when messages are exchanged between two parties. The design of the HMAC standard also allows for non-repudiation protection as described in 2.1.d.

CIAN Service: The HMAC protocol provides for Authentication and Confidentiality of shared secret $[A_0C_0]$. However, since a message digest can also be included, it can be used to sign a transaction, i.e. provide Non-repudiation service $[N_0]$.

c. Early Integrity checking algorithms.

Several checksum based integrity protection algorithms have been standardized and used over many years. As these early protection mechanisms typically involve simple types of logic operations, they are often embedded into hardware or firmware products to operate at line-speed.

- Parity.

Since the RS/IEA232 standard was defined and fielded in the early 1960's, [7] parity checking has been used mostly on computer/terminal communications links (and computer memory). However, as encapsulation protocols like PPP (RFC-1172) took hold, or as data links were shortened to a few feet (e.g. terminal/printer to local computer connection), the parity protection was frequently turned off.

CIAN Service: The Parity protocol only provides Integrity protection [I_0]

- Cyclic Redundancy Check (CRC).

The CRC Integrity checking was originally developed to provide for efficient integrity verification in various hardware and firmware applications. [8] Today, it is still used in many block communication protocols and hard-disk I/O sub-assemblies.

CIAN Service: The CRC protocol only provides Integrity protection [I_0].

- Hamming Code.

The Hamming Code [9] was developed to provide Integrity detection as well as automated recovery for single bit errors. Hamming code is typically used in one-way data-links or in applications where error detection and subsequent retransmission is not effective (e.g. in satellite links).

CIAN Service: The Hamming code provides Integrity protection [I_0] only.

3.2 Single-Key Crypto Solutions.

a. Data Encryption Standard (DES/TDES).

Since the development and release of the Data Encryption Standard (DES) [10] in 1977, the DES algorithm has been used in countless applications. Only after it was demonstrated the applied 56-bit DES key could be compromised, using exhaustive-forced attacks, the Tripple DES (TDES) algorithm was added to the standard in the late 1990's. The TDES performs three successive encryption/decryption operations using three different (56-bit) key-sets.

CIAN Service: The DES/TDES algorithm provides Confidentiality of all exchanged data [C_1].

b. Advance Encryption Standard (AES).

After an open competition to develop the successor for DES/TDES, the winning Advanced Encryption Standard (AES -- also known as the Rijndael algorithm), was adopted as FIPS standard 197. [11] The AES operates with 128, 192, and 256 bit keys, which should make it safe for many years to come.

CIAN Service: The AES algorithm provides Confidentiality of all exchanged data [C_1].

c. International Data Encryption Algorithm (IDEA) [C_1].

The International Data Encryption Algorithm (IDEA) [12] was developed in the early 1990's and released for non-profit use in popular "open" products like Pretty Good Privacy (PGP). It has since then also been accepted for use in Secure Multimedia Internet Mail Extension (S/MIME) and Cryptographic Message Syntax (CMS) message formats (RFC-3058).

CIAN Service: The IDEA provides Confidentiality of all exchanged data [C_1].

d. Kerberos Protocol.

Since the initial release of the Kerberos authentication protocol by the Massachusetts Institute of Technology (MIT), it has undergone several revisions and is now available as version five (RFC-1510). Today, it is primarily used for authentication over an open network but can also provide limited Confidentiality service. After Microsoft adapted Kerberos for use with it's win2000 and winXP operating systems, and after it has been expanded to support TDES encryption, Kerberos has enjoyed an increasing amount of popularity.

CIAN Service: The Kerberos protocol provides Authentication, and Confidentiality of exchanged credentials [A_1C_1].

3.3 Dual-Key Crypto Solutions.

a. RSA Algorithm.

The Rivest Shamir Adleman (RSA) dual-key algorithm was invented in 1977 by Ronald L. Rivest, Adi Shamir, and Leonard Adleman. [13] While the RSA algorithm initially was available under license from RSA Security only, it has been adopted widely by various industries, including the banking industry, which published it as standard X9.31.

As the RSA patent was about to expire (happened September 8, 2000) NIST approved the RSA algorithm for use in the Federal Government. [14] The RSA algorithm is designed to support enciphering using the private key (i.e. signing) as well as the public-key (i.e. encryption).

CIAN Service: The public RSA key can be used to provide for Confidentiality of encrypted data [C_2]. The private RSA key can also be used for encryption, hereby providing for signature and Non-repudiation service [N_2].

b. DSS Algorithm.

In 1994 NIST published the Digital Signature Standard (DSS) as FIPS-186. [15] It is also known as the Digital Signature Algorithm (DSA). Since the Government at that time was still "experimenting" with other dual-key encryption algorithms (as an alternative to using RSA), the DSS algorithm was restricted for use with signature-only applications.

CIAN Service: Only the private DSA key can be used for encryption, hereby limiting DSA to provide signature and Non-repudiation service only [N_2].

c. Diffie-Hellman Key Exchange Agreement.

The Diffie-Hellman key exchange agreement (RFC-2631) can be used to generate a shared secret to be used between two parties (over untrusted networks) after they have both generated a dual-key set and exchanged their respective public-key.

As all public information for a given user can be retrieved from an existing DSS/DSA X.509 certificate, the Diffie-Hellman key-exchange can also be used in an off-line mode, for example when exchanging e-mail messages encrypted with a single session key.

CIAN Service: The Diffie-Hellman key-exchange derives a shared session key while providing full confidentiality [C_2].

d. SSL/TLS Protocol.

The Secure Socket Layer (SSL) protocol, developed by Netscape, [16] is today one of the most successful implementations offering strong Authentication as well as strong Confidentiality. While SSL based client-authentication is supported (true for most versions of the Netscape and Microsoft browsers), it is today used mostly to provide strong authentication of WEB servers. However there are a few WEB-services (several are Government operated) that will force the user/client to select a certificate (and matching private key) that is required to perform client-authentication before access to specific services are granted.

In 1999, the Internet Engineering Task Force (IETF) network working group published a derivative of SSL also knows as the Transport Layer Security (TLS) protocol (RFC-2246). Although differences between the two protocols are minor, they will not interoperate. Only time will tell which of the two protocols will dominate in the future.

CIAN Service: The SSL/TLS protocol is using a single-key encryption algorithm to provide Integrity protection [C_1]. Strong authentication of server and/or client entity is provided as an option [A_2].

e. SAML Protocol.

The Security Association Markup Language (SAML) is the latest authentication type protocol in the market place. [17] It is defined for use in the Extensible Makeup Language

(XML) WEB services community and it can support a variety of authentication methods like, password, Kerberos, Secure Remote Password (SRP), hard-token, SSL/TLS, public key from various formats of certificates, and XML signature. The SAML definitions will not only carry authentication information but can also include various assertions and attributes associated with the authenticated user.

CIAN Service: The SAML protocol provides for several optional Authentication services, which (among others) are based on, Password [A_0], Kerberos [A_1], and public key certificates [A_2].

f. S/MIME Protocol.

Although hampered by the sluggish deployment of PKI infrastructures, the S/MIME e-mail protocol (RFC-2633) is slowly put in use to provide a complete set of strong CIAN Security Solutions.

CIAN Service: The S/MIME protocol provides for Integrity and Confidentiality using hash algorithms [I_0] and single-key encryption [C1]. Authentication and Non-repudiation is provided using dual-key signature algorithms [A_2N_2].

g. IPSec protocol.

CIAN Service: The Internet Protocol Secure (IPSec) protocol supports strong authentication using pre-shared keys [A_1] or X.509v3 certificates [A_2] (RFC-2411). Confidentiality is provided via a secure tunnel encryption algorithm using a single-key [C_1]. Integrity protection is provided using a hash algorithm [I_0].

4. Summary.

Table 2-1 clearly shows that Confidentiality, Authentication, and Non-repudiation services can be provided using none, one or two keys (along with the appropriate hash, encrypt, or decrypt algorithm). Integrity, however, should only be implemented using a hash (no-key) algorithm. Although Table 2-1 indicates basic Confidentiality, Authentication, and Non-repudiation services can be provided using all three key configurations, the table legends identify several differences in the benefits and drawbacks that apply to each combination.

All of the standards analyzed in paragraph three provide one or more of the CIAN and key combinations shown in Table 2-1. The most complex security standards (e.g. 3.3-e, 3.3-f, 3.3-g) are typically categorized using multiple CIAN and key combinations.

The above-applied methods for analyzing basic CIAN security services can also be applied to other standards or products. Once the supported CIAN solution(s) and the applied key-combination are determined, Table 2-1 (and the text in paragraph two) can

be used to derive all benefits and drawbacks that apply to a particular standard or product under consideration.

This paper concludes the number of keys used, when embedding cryptographic functions in a given system or product, appears to have a significant influence on the resulting strength and nature of the security design. A specific key configuration (i.e. the column in Table 2-1) is, therefore, one of the first security design steps to be undertaken when starting on a new system/product development effort. The actual CIAN security solutions to be provided are more likely to be specified in (or derived from) existing system/product requirements.

The paper also suggests the selection of the key-configuration will in most cases also define the type of cryptographic algorithm(s) that can be applied successfully. This order of selection appears to be simpler (only three choices) than trying first to select a given algorithm or protocol from a relative high number of possible options.

The final choice of a specific algorithm (or protocol) is easier, when the type of algorithm/protocol is already given. For example, choosing between a DES, TDES, and AES algorithm is fairly easy considering known market-segment, support, and life cycle parameters.

5. References.

1: Schneier, Bruce. Applied Cryptography Second Edition. New York: John Wiley, 1996, Ref. Paragraph 3.2).

2: US 106th Congress. "Electronic Records And Signatures in Commerce." Public Law 106-229. June 30, 2000. URL: http://frwebgate.access.gpo.gov/cgi-bin/getdoc.cgi?dbname=106_cong_public_laws&docid=f:publ229.106.pdf (5-24-2003).

3: Schneier, Bruce. Applied Cryptography Second Edition. New York: John Wiley, 1996, Ref. Paragraph 2.2).

4: Schneier, Bruce. Applied Cryptography Second Edition. New York: John Wiley, 1996, Ref. Paragraph 2.5).

5: Internet Engineering Task Force. "Request For Comments". URL: http://www.ietf.org/rfc.html (5-24-2003)

6: Federal Information Processing Standard. "Secure Hash Standard". FIPS PUB 180-1. April 17, 1995. URL: http://www.itl.nist.gov/fipspubs/fip180-1.htm (5-24-2003).

7: Strangio E. Christopher. "The RS232 Standard". 1997. URL: http://www.camiresearch.com/Data_Com_Basics/RS232_standard.html (5-24-2003).

8: Rebel Eric-Paul. “Cyclic Redundancy Check”. 2000-09-05. URL: http://utopia.knoware.nl/users/eprebel/Communication/CRC/ (5-24-2003).

9: University of New Brunswick. “Error Correction and the Hamming Code” EE4253 Digital Communications. April 02. 2002. URL: http://www.ee.unb.ca/tervo/ee4253/hamming.htm (5-24-2003).

10: U.S. Department of Commerce. “Data Encryption Standard”. FIPS PUB 46-3. October 25, 1999. URL: http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf (5-24-2003).

11: Federal Information Processing Standard. “Advanced Encryption Standard (AES)”. FIPS PUB 197. November 26, 2001. URL: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf (5-24-2003).

12: X. Lai, "On the design and security of block ciphers", ETH Series in Information Processing, J.L. Massey (editor), vol. 1, Hartung-Gorre Verlag Konstanz, Technische Hochschule (Zurich), 1992.

13: RSA Security. “RSA-based Cryptographic Schemes”. URL: http://www.rsasecurity.com/rsalabs/rsa_algorithm/index.html (5-24-2003).

14: NIST. “Digital Signature Standard (DSS)”. Now supporting DSA, RSA and EVDSA. November 15, 2002. URL: http://csrc.nist.gov/cryptval/dss.htm (5-24-2003).

15: Federal Information Processing Standard. “Digital Signature Standard (DSS)”. FIPS PUB 186. May 19, 1994. URL: http://www.itl.nist.gov/fipspubs/fip186.htm (5-24-2003).

16: Netscape. “SSL 3.0 Specification”. URL: http://wp.netscape.com/eng/ssl3/ (5-24-2003).

17: OASIS. “Assertion and Protocol for OASIS Security Assertion Markup Language (SAML)”. November 5, 2002. URL: http://www.oasis-open.org/committees/download.php/1371/oasis-sstc-saml-core-1.0.pdf (5-24-2003).

Last Updated: July 25th, 2017

Upcoming Training

SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
Community SANS Omaha SEC401*	Omaha, NE	Aug 14, 2017 - Aug 19, 2017	Community SANS
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
Virginia Beach 2017 - SEC401: Security Essentials Bootcamp Style	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Pasadena SEC401 @ NASA	Pasadena, CA	Aug 23, 2017 - Aug 30, 2017	Community SANS
Mentor Session - SEC401	Minneapolis, MN	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
Mentor Session - SEC401	Edmonton, AB	Sep 06, 2017 - Oct 18, 2017	Mentor
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Mentor Session - SEC401	Ventura, CA	Sep 11, 2017 - Oct 12, 2017	Mentor
Community SANS Albany SEC401	Albany, NY	Sep 11, 2017 - Sep 16, 2017	Community SANS
Community SANS Dallas SEC401	Dallas, TX	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Columbia SEC401	Columbia, MD	Sep 18, 2017 - Sep 23, 2017	Community SANS
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Boise SEC401	Boise, ID	Sep 25, 2017 - Sep 30, 2017	Community SANS
Baltimore Fall 2017 - SEC401: Security Essentials Bootcamp Style	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS New York SEC401	New York, NY	Sep 25, 2017 - Sep 30, 2017	Community SANS
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Charleston SEC401	Charleston, SC	Oct 02, 2017 - Oct 07, 2017	Community SANS
Mentor Session - SEC401	Arlington, VA	Oct 04, 2017 - Nov 15, 2017	Mentor
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event