Algorithm Validation Testing

Larry Bassham

and

Sharon Keller

Validation Testing for Cryptographic Algorithms

- Prerequisite to FIPS 140 Validation Testing

Validation Process

- NIST and CSE develop validation suite consisting of validation tests
- Accreditation Laboratory supplies data and applicable validation tests required to validate a vendor’s Implementation Under Test (IUT)
- Vendor runs the specified validation tests on the IUT and returns the results to the laboratory
- Laboratory verifies results and sends the IUT’s results and a request for validation to NIST

Validation Process (Continued)

- NIST reviews the results
  - Adds the vendor algorithm IUT information to NIST's algorithm validation database
  - Creates an algorithm validation certificate which NIST and CSE sign and send to the laboratory to be sent to the vendor
  - Adds an entry to the appropriate validation list which is accessible via http://csrc.nist.gov/cryptval

Cryptographic Algorithms for Which NIST Currently Has Standards and Validation Tests

- FIPS 197 – *Advanced Encryption Standard (AES)*
- FIPS 46-3 and FIPS 81 – *Data Encryption Standard (DES)* and *DES Modes of Operation* – specifies the DES and Triple DES algorithms
- FIPS 186-2 and FIPS 180-1 – *Digital Signature Standard (DSS)* and *Secure Hash Standard (SHS)*
- FIPS 185 – *Escrowed Encryption Standard (EES)* – specifies the Skipjack algorithm

Cryptographic Algorithms for Which NIST Is Developing Standards and Validation Tests

- HMAC
- SHA-256, SHA-384, SHA-512
- Key Establishment using Diffie-Hellman and MQV

Validation Tests (TDES, DES, and Skipjack)

- 3 types of cryptographic algorithm validation tests
  - Known Answer tests
    - Variable Plaintext/Ciphertext Known Answer Test
    - *Inverse Permutation Known Answer Test
    - Variable Key Known Answer Test
    - *Permutation Operation Known Answer Test
    - *Substitution Table Known Answer Test
  - *Multi-block Message test
  - Monte Carlo test

(Not run for Skipjack)*

Validation Tests (TDES, DES, and Skipjack) (Continued)

- A separate set of Known Answer Tests and a corresponding Monte Carlo test and (if applicable) a Multi-block Message Test exist for every state in every mode of operation

Validation Tests (TDES, DES, and Skipjack) (Continued)

- States
  - Encrypt
  - Decrypt

Validation Tests (TDES, DES, and Skipjack) (Continued)

Modes of Operation per Algorithm		
DES	**Triple DES**	**Skipjack**
ECB	ECB	ECB
CBC	CBC	CBC
CFB - 1,8,64 bit	CBC-Interleaved	CFB - 1,8,64 bit
OFB	CFB - 1,8,64 bit	OFB
	CFB-Pipelined - 1,8,64 bit	
	OFB	
	OFB-Interleaved	

Known Answer Tests (Triple DES and DES)

- Verifies that the implementation correctly performs the algorithm
- Provides conformance testing for components of the algorithm
  - Initial permutation IP
  - Expansion matrix E
  - Inverse permutation IP^{-1}
  - Key permutation PC1 and PC2
  - Data permutation P
  - Substitution tables S_1, S_2, …, S_8

Known Answer Tests
(Skipjack)

- Verifies that, given known inputs, the correct results are produced.

Known Answer Tests

- Test procedures:
  - Lab supplies known values for key(s), plaintext, and, if applicable, IV(s) for every Known Answer Test
  - Vendor runs each known value through the IUT of the TDES, DES, or Skipjack algorithms
  - The results are recorded and compared to the known answers

Multi-block Message Test

- Tests the ability to properly process multi-block messages, requiring the chaining of information from one block to the next
- Lab supplies the IUT with pseudorandom values for key(s), messages that are integral numbers of blocks in length, and, if applicable, IV(s)
- Evaluates the resulting ciphertext

Monte Carlo Test

- Tests for implementation flaws
- Lab supplies pseudorandom values for key(s), plaintext, and, if applicable, IV(s)
- Test consists of 4 million cycles through the TDES, DES, or Skipjack algorithms
- The results of every 10,000th encryption or decryption cycle are recorded and evaluated

Validation Tests (AES)

- Same three test types as DES and 3DES
  - Know Answer Tests
  - Multi-block Message Test
  - Monte Carlo Test
- Known Answer Tests differ
  - Variable Plaintext/Ciphertext Test
  - Variable Key Test
  - GFSbox Test (stress math operations and Sbox used in round function
  - Key Sbox Test (stresses Sbox used in key schedule)
- Separate tests required for each of three key sizes as well as each state (encrypt/decrypt)

SHA-1

- Three Tests
  - Short Messages
  - Selected Long Messages
  - Monte Carlo
- Two modes: bit-oriented and byte-oriented

SHA-1: Short Messages

- Tests the ability of the IUT to generate digests for messages of arbitrary length
- Generates messages of length $0 \leq i \leq 1024$
- Calculates message digests of messages
- Compares message digest calculated with those supplied by the tool

SHA-1: Selected Long Messages

- Tests the ability of the IUT to generate digests for messages that span multiple blocks
- Generates messages of length:
  - Byte-oriented: $1032+i*1024$, $0 \le i < 100$
  - Bit-oriented: $1025+i*1024$, $0 \le i < 100$
- Calculates message digests of messages
- Compares messages digests calculated with those supplied by the IUT

SHA-1: Monte Carlo

- Tests for implementation flaws by providing pseudo-random input
- Generates 100 message digests, using previous digests as input
- Compares messages digests calculated by the IUT with the expected values

DSS

- Primality
- Domain Parameter Generation
- Domain Parameter Validation
- Key Pair Generation
- Signature Generation
- Signature Validation

Primality

- Tests the ability of the implementation to determine whether large numbers are prime
- Generates several large numbers, some of which are prime and some of which are not
- Compares the determination of the IUT regarding primality with the expected value

Domain Parameter Generation

- Tests the ability of the IUT to generate Domain Parameters
- Requests the IUT to generate a specific number of Domain Parameter sets (P, Q, G, Seed, Counter, H)
- Recalculates the Domain Parameters using the same Seed and verifies the remaining values in the set

Domain Parameter Validation

- Tests the ability of the IUT to recognize valid vs. invalid Domain Parameters
- Generates several sets of Domain Parameters
- Modifies components of some parameters
- Compares the IUT's results with the expected values

Key Pair Generation

- Tests the ability of the IUT to generate pair-wise consistent keys
- Provides seed values to the IUT to use to generate key pairs
- Verifies the consistency of key pair or performs public key validation on the public key

Signature Generation

- Tests the ability of the IUT to derive correct signatures for messages
- Generates several messages to be signed by the IUT
- If the IUT can output the private key, compares the derived signatures with the expect signatures
- Otherwise, runs signature verification

Signature Verification

- Tests the ability of the IUT to recognize valid vs. invalid signatures
- Generates several messages and signatures
- Alters some signatures or messages to introduce errors
- Compares the IUT’s results with the expected values

Questions??