Low-Cost Self-Test of Crypto Devices

G. DiNatale, M. Doulcier, M-L. Flottes, B. Rouzeyre

WDSN 2008
Motivation

- Secure circuits testing
  - Scan path
    - High fault coverage
    - Automatic generation of scan chains
    - Easy test sequence generation

- Vulnerability
  - Control and observation of internal states of CUT
  - => secret data retrieval

Scan based attack DES [Yan et al., ITC 04]
AES [Yan et al., IEEE TCAD 06]
Motivation

- BIST
  - Reduced ATE cost
  - In-situ testing
  - Reduced external access

- But
  - Circuitry overhead
    - test controller
    - pattern generator
    - signature analyzer…
Secure circuits contain a crypto core
E.g. Smart cards

- Crypto core => Test resource
Outline

- AES & DES
  - Algorithm & architecture
  - Testability issues
- AES/DES as pattern generators
- AES/DES Self test
- Optimisations
- Conclusion
Symmetric cryptography

- **DES**
  - Adopted as standard in 1976
  - Data: 64 bits, Key: 56 bits

- **AES: Advanced Encryption Standard**
  - Adopted as standard in 2001
  - Data: 128 bits, Key: 128 bits (192, 256)

Crypto algorithms basis: Diffusion & Confusion
Diffusion and confusion

- **Confusion** refers to making the relationship between the key and the ciphertext as complex and involved as possible.
- **Diffusion** refers to the property that redundancy in the statistics of the plaintext is "dissipated" in the statistics of the ciphertext. For diffusion to occur a change in a single bit of the plaintext should result in changing the value of many ciphertext bits.

Iterative algorithms (rounds)

- Each round is a "bijective" operation
**DES algorithm & architecture**

- **Plaintext (64 bits)**
  - IP
  - F
  - F
  - F
  - F
  - FP
  - Ciphertext (64 bits)

- **Half Block (32 bits)**
  - E
  - S1, S2, S3, S4, S5, S6, S7, S8
  - P

- **Subkey (48 bits)**
  - Control
  - FP
  - Register R1
  - Cipher text

- **Key Generation**
  - Start
Diffusion
- every input bit of a round influences many output bits, i.e. every input line of a round is in the logic cone of many output bits.
- an error caused by a fault in the body of the round is very likely to propagate to the output.
- observability

Bijective
- controllability

Highly testable hardware implementations
- => random testing
One test pattern = Intermediate round result of encryption
AES/DES as TPG: randomness analysis

NIST Special Publication 800-22

<table>
<thead>
<tr>
<th>Test Number</th>
<th>Test Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Monobit Test</td>
</tr>
<tr>
<td>2</td>
<td>Block Frequency Test</td>
</tr>
<tr>
<td>3</td>
<td>Cumulative Sums Forward (Reverse)</td>
</tr>
<tr>
<td>4</td>
<td>Runs Test</td>
</tr>
<tr>
<td>5</td>
<td>Long Runs of Ones Test</td>
</tr>
<tr>
<td>6</td>
<td>Rank Test</td>
</tr>
<tr>
<td>7</td>
<td>Discrete Fourier Transform (Spectral) Test</td>
</tr>
<tr>
<td>8</td>
<td>Universal Statistical Test</td>
</tr>
<tr>
<td>9</td>
<td>Approximate Entropy Test</td>
</tr>
<tr>
<td>10</td>
<td>Serial Test</td>
</tr>
<tr>
<td>11</td>
<td>Linear Complexity Test</td>
</tr>
<tr>
<td>12</td>
<td>Aperiodic Templates</td>
</tr>
<tr>
<td>13</td>
<td>Periodic Template Test</td>
</tr>
<tr>
<td>14</td>
<td>Random Excursion Test</td>
</tr>
<tr>
<td>15</td>
<td>Random Excursion Variant Test</td>
</tr>
</tbody>
</table>

Statistical package of 15 tests has been developed to test binary sequences randomness
1.5 Mbit bitstream (leftmost bit)

Test passes if $x > 0.1$

<table>
<thead>
<tr>
<th></th>
<th>1_round AES</th>
<th>1_round DES</th>
<th>LFSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>0.71209</td>
<td>0.45847</td>
<td>0.00256</td>
</tr>
<tr>
<td>Blk-freq</td>
<td>0.47556</td>
<td>0.87065</td>
<td>0.44150</td>
</tr>
<tr>
<td>Runs</td>
<td>0.64156</td>
<td>0.18337</td>
<td>0.14362</td>
</tr>
<tr>
<td>Long Runs</td>
<td>0.28546</td>
<td>0.15829</td>
<td>0.96593</td>
</tr>
<tr>
<td>Rank</td>
<td>0.35722</td>
<td>0.24411</td>
<td>0.52660</td>
</tr>
<tr>
<td>DFT</td>
<td>0.03397</td>
<td>0.61040</td>
<td>0.81051</td>
</tr>
<tr>
<td>Aperiodic</td>
<td>0.50704</td>
<td>0.50541</td>
<td>0.49963</td>
</tr>
<tr>
<td>Periodic</td>
<td>0.08345</td>
<td>0.90055</td>
<td>0.39384</td>
</tr>
<tr>
<td>Univ.Maurer</td>
<td>0.44635</td>
<td>0.86625</td>
<td>0.24403</td>
</tr>
<tr>
<td>Lincomp</td>
<td>0.86761</td>
<td>0.88996</td>
<td>0</td>
</tr>
<tr>
<td>Serial</td>
<td>0.62350</td>
<td>0.42735</td>
<td>0.71383</td>
</tr>
<tr>
<td>Apen</td>
<td>0.44173</td>
<td>0.41358</td>
<td>0.63747</td>
</tr>
<tr>
<td>Cusum</td>
<td>0.73566</td>
<td>0.55751</td>
<td>0.00326</td>
</tr>
<tr>
<td>Random</td>
<td>0.41284</td>
<td>0.36790</td>
<td>0</td>
</tr>
<tr>
<td>Variant-R</td>
<td>0.49847</td>
<td>0.24177</td>
<td>0</td>
</tr>
</tbody>
</table>
Proportion of bitstreams passing each NIST test

1-round AES

LFSR

randomness:

“1-round AES” ≈ “1-round DES” ≈ LFSR
AES Self-test

Cycle 1

Secret Key K

Message

Start

0 MUX 1

0

Control

Sub Bytes

Shift Row

Mix Column

MUX

RK_i

Key Generation

Key

Last-round

Register R1

Done

Round

Last-round

Key

0

Start

0 MUX 1
AES Self-test

Cycle 2, 3, ……, T

Secret Key K

Start

Control

Round

Key

Last-round

Key Generation

Done

0 MUX

1

Sub Bytes

Shift Row

Mix Column

MUX

RK$_i$

Register R1

Signature

Is FC = 100% achievable ?

When ?
AES Self-test

SubBytes

ShiftRows

MixColumns

AddKey

Register
AES Self-test

Sbox (8 bits → 8 bits)

Implementations
- ROM => 256 patterns
- Glue logic => 200 ... 220 patterns
- Actually 203 patterns
AES Self-test

One Sbox (8 bits → 8 bits)
- Glue logic => 203 patterns
- 203 responses

- Propagation of Sboxes errors
- Faults in Mixcolumn, Addkey, Register are tested by the 203 responses
AES Self-test

- How many random patterns are needed to get those 203 deterministic patterns? “The Coupon Collector Problem”

\[ P(X_1 \cap X_2 \cap \ldots \cap X_k) = 1 - \sum_{j=1}^{k} (-1)^{j-1} \binom{j}{k} \left( \frac{m-j}{m} \right)^T \]

- \( m = 2^{128} \)
- \( k = \# \text{vectors} = 203 \)

\[ P(X_1 \cap X_2 \cap \ldots \cap X_k) = 99\% \quad \Rightarrow \quad T = 2534 \text{ random patterns} \]

- Sbox implementation:
  - #test vectors \( \in \{200, \ldots, 256\} \Rightarrow T \in \{2520, \ldots, 2590\} \]
“Pseudo” Fault Simulation

- Result:
  - Fault coverage: 100% after 2534 cycles
  - Test time reduction: 2400 cycles (with several keys, several plaintexts)

- Specific plaintext, specific key for minimal test time?
DES Self-test

Right bits

32 bits

Expansion

Sbox (6 bits → 4 bits)

- Propagation of Sboxes errors

- Faults in Addkey, Permutation, Expansion & Register are tested by the 64 responses

Sbox (6 bits)

Actually 64 patterns

Expansion

Permutation

Key

Left bits
DES random sequence length

\[ p(X_1 \cap X_2 \cap \ldots \cap X_k) = 1 - \sum_{j=1}^{k} (-1)^{j-1} \binom{k}{j} \left( \frac{m-j}{m} \right)^T \]

\[ m = 2^{64} \quad \text{k = #vectors = 64} \]

\[ p(X_1 \cap X_2 \cap \ldots \cap X_k) = 99\% \]

\[ T = 540 \quad \text{random patterns (540 rounds)} \]

\[ \text{34 encryptions} \]

Results: 100% FC after 24 encryptions (Data path and control)
- Speeding up self-test of AES
  - 2500 cycles for 256 test patterns
  - Feed-back on Sbox

  - 5 cycles in state graph =>

  - Add a (simple) feed-back function for traversing all 256 states
    \[ g = \text{exor} (01110110) \rightarrow 5 \text{ inverters} \]
- 2 steps procedure
  - test of Sboxes: 256 cycles (vs 2400)
  - test of remaining logic: 16 cycles

- Area overhead: 1%
Conclusion

- AES/DES as TPG
  - Randomness: better than LFSRs

- Self Testability
  - AES: 2400 encryption rounds (of a single message)
  - DES: 540 encryption rounds (of a single message)
  - Suitable technique for other ciphering circuits (IDEA, Fox, Blowfish, ...)

  - No area overhead
  - No impact on performance
  - No impact on security
References

- [FIPS PUB 46-3]: DATA ENCRYPTION STANDARD (DES), 1999 October 25
- [http://www.commentcamarche.net/crypto/des.php3]
- [FIPS PUB 197]: Announcing the ADVANCED ENCRYPTION STANDARD (AES), 2001 November 26
- [Sch97]: B. Schneier, Cryptographie appliquée : protocoles, algorithmes et codes sources en C, J. Wiley, 1997 (p491-499)
- [http://www.securiteinfo.com/crypto/cracked.shtml]
- [Yan04]: B. Yang, K. Wu, R. Karri, Polytechnic University, "Scan-based Side-Channel Attack on Dedicated Hardware Implementations on Data Encryption Standard", International Test Conference (ITC 2004), Charlotte, USA, October 26-28, pp 339-344
- [Yan05]: B. Yang, K. Wu and R. Karri, Secure Scan: A Design-for-Test Architecture for Crypto Chips, Design Automation Conference (DAC 2005), Anaheim, July 12-14, pp 135-140, 2005
- [Yan, FDTC 05]: B. Yang & R. Karri, "Crypto BIST: A Built-In Self Test Architecture for Crypto Chips", 2nd Workshop on fault diagnosis and tolerance in cryptography (FDTC 2005), pp 95-108
- [NIST 800-22]: A statistical test suite for random and pseudorandom number generators for cryptographic applications NIST Special Publication 800-22 (with revisions dated May 15, 2001)
- **Monobit Test**: determine whether the number of ones and zeros in a sequence are approximately the same as would be expected for a truly random sequence.

- **Block Frequency Test**: determine whether the number of ones and zeros in each of M non-overlapping blocks created from a sequence appear to have a random distribution.

- **Cumulative Sums Forward (Reverse) Test**: determine whether the sum of the partial sequences occurring in the tested sequence is too large or too small.

- **Runs Test**: determine whether the number of runs of ones and zeros of various lengths is as expected for a random sequence. In particular, this test determines whether the oscillation between such substrings is too fast or too slow.

- **Long Runs of Ones Test**: determine whether the longest run of ones within the tested sequence is consistent with the longest run of ones that would be expected in a random sequence.

- **Rank Test**: check for linear dependence among fixed length substrings of the original sequence.

- **Discrete Fourier Transform (Spectral) Test**: detect periodic features (i.e., repetitive patterns that are near each other) in the tested sequence that would indicate a deviation from the assumption of randomness.

- **Aperiodic Templates Test**: reject sequences that exhibit too many occurrences of a given non-periodic (aperiodic) pattern.

- **Periodic Template Test**: reject sequences that show deviations from the expected number of runs of ones of a given length.

- **Universal Statistical Test**: detect whether or not the sequence can be significantly compressed without loss of information. A compressible sequence is considered to be nonrandom.

- **Approximate Entropy Test**: compare the frequency of overlapping blocks of two consecutive/adjacent lengths (m and m+1) against the expected result for a normally distributed sequence.

- **Random Excursion Test**: determine if the number of visits to a state within a random walk exceeds what one would expect for a random sequence.

- **Random Excursion Variant Test**: detect deviations from the distribution of the number of visits of a random walk to a certain state.

- **Serial Test**: determine whether the number of occurrences of m-bit overlapping patterns is approximately the same as would be expected for a random sequence.

- **Linear Complexity Test**: determine whether or not the sequence is complex enough to be considered random.