Scan chain encryption in Test Standards
Mathieu da Silva, Giorgio Di Natale, Marie-Lise Flottes, Bruno Rouzeyre

To cite this version:

HAL Id: lirmm-01882578
https://hal-lirmm.ccsd.cnrs.fr/lirmm-01882578v2
Submitted on 10 Oct 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
SCAN CHAIN ENCRYPTION IN TEST STANDARDS

Mathieu Da Silva, Marie-Lise Flottes, Giorgio Di Natale, Bruno Rouzeyre

SURREALIST 2018
**CONTEXT**

- Test standards
  - IEEE 1149 (JTAG) for board testing
  - IEEE 1500 for cores testing in a SoC
  - IEEE 1687 (IJTAG) for embedded instruments
**CONTEXT**

- **Threats**
  - Untrusted devices
    - *Rosenfeld et al., Attacks and Defenses for JTAG, IEEE Design & Test 2010*
  - Malicious users
    - (example: scan attacks)
    - *Yang et al., Secure Scan: A Design-for-Test Architecture for Crypto Chips, TCAD’06*


**SUMMARY**

1) **Scan chain encryption**

2) **State-of-the-art based on test communication encryption**

3) **Implementation with block cipher**

4) **Implementation with stream cipher**

5) **Conclusion**
SCAN CHAIN ENCRYPTION

- Solution: test communication encryption

- **Input decryption** prevents sending desired test data
- **Output encryption** prevents reading plain test responses
SCAN CHAIN ENCRYPTION

Solution: test communication encryption

- Input decryption prevents sending desired test data
- Output encryption prevents reading plain test responses
- Test/debug only possible by authorized user knowing the secret key
2 types of symmetric cipher: stream and block ciphers
STREAM CIPHER / BLOCK CIPHER

- **Stream cipher encryption**
  - Keystream XORed **bitwise** with the plaintext

- **Block cipher encryption**
  - Confusion and diffusion on a **block** of plaintext

- **Preference for stream ciphers**
  - "Naturally" adapted to serial test communication (JTAG, IEEE 1500, IJTAG)
  - Smaller area footprint compared to block ciphers
  - But ..
**Two-times pad: stream cipher requirement**

- **Two-times pad**: same key and IV re-used => same keystream generated to encrypt different data

⇒ Possible to carry out attacks if requirement is not fit

\[ R_1 \oplus S(IV, Key) \oplus R_2 \oplus S(IV, Key) \]

⇒ Solution: IV generated randomly at each circuit reset

\[ R_1 \oplus S(IV_1, Key) \oplus R_2 \oplus S'(IV_2, Key) \]
SUMMARY

1) Scan chain encryption

2) State-of-the-art based on test communication encryption

3) Implementation with block cipher

4) Implementation with stream cipher

5) Conclusion
STREAM-BASED ENCRYPTION ON JTAG INTERFACE

- Challenge/Response protocol to encrypt JTAG test communication

1) Challenge \( C \)

2) Response as Key

Key

\[ S(IV, Key) \]

3) Encryption of the JTAG TDR with the keystream \( S(IV, Key) \)

Rosenfeld et al., Attacks and Defenses for JTAG, IEEE Design & Test 2010

Requirement not fulfilled

01/06/2018
STREAM-BASED ENCRYPTION ON IEEE 1500 INTERFACE

- IEEE 1500 standard
  - Similar as JTAG standard, but for SoC wrappers
  - Parallel test inputs WPI and parallel test outputs WPO
STREAM-BASED ENCRYPTION ON IEEE 1500 INTERFACE

- Encrypt test data on a targeted core (IEEE 1500)
  1) Send the key to the core via specific scan chain non-visible from the others cores

Rosenfeld et al., Security-Aware SoC Test Access Mechanisms, VTS’11

01/06/2018
**Stream-based Encryption on IEEE 1500 Interface**

- Encrypt test data on a targeted core (IEEE 1500)
  1. Encrypt the parallel input/output (WPI and WPO)

Rosenfeld et al., *Security-Aware SoC Test Access Mechanisms, VTS'11*

01/06/2018
STREAM-BASED ENCRYPTION ON IJTAG INTERFACE

- Encryption of Test Data Register associated to Instruments in the IJTAG network

Kan et al., Echeloned IJTAG data protection, AsianHOST 2016.

01/06/2018
OUR PROPOSITION

- Insertion of block or stream ciphers at Scan-In and Scan-Out

- Assumption: original circuit embedded a crypto-core with its key management and storing

- Scan chain encryption solution shares the key management and storing already implemented
1) Scan chain encryption

2) State-of-the-art based on test communication encryption

3) Implementation with block cipher

4) Implementation with stream cipher

5) Conclusion
**Block Cipher-based Scan Encryption**

- Implementation on scan chain with 2 PRESENT block ciphers:
  - Lightweight (1 PRESENT = 2 139 GE)
  - Encryption by 64-bits block size
MODE OF OPERATIONS

- 64 bits encrypted every 32 clock cycles

\[ S_i \quad (64 \text{ bits}) \quad S_{i-1} \quad (64 \text{ bits}) \quad S_2 \quad (64 \text{ bits}) \quad S_1 \quad (64 \text{ bits}) \]

\[ \Rightarrow \ #SFF = P \times 64 \]

\[ \Rightarrow \text{No test time overhead on each pattern} \]
MODE OF OPERATIONS

- U bits = Unused bits

$S_1$

R+U = 64 bits

R=#SFF mod 64

U bits added

$\Rightarrow \#SFF = Px64 + R$

$\Rightarrow$ Loss of U clock cycles per pattern
SUMMARY

1) Scan chain encryption

2) State-of-the-art based on test communication encryption

3) Implementation with block cipher

4) Implementation with stream cipher

5) Conclusion
STREAM CIPHER-BASED SCAN ENCRYPTION

- Implementation on JTAG:
  - 1 TRIVIUM stream cipher (2 016 GE)
  - TRNG to generate random IV
  - New instruction GetIV with a test data register IV

- Mode of operations in 2 phases: initialization and encryption
INITIALIZATION PHASE

1) TRNG initialization: reach sufficient entropy to generate random number
INITIALIZATION PHASE

2) Shift IV in the dedicated Test Data Register
INITIALIZATION PHASE

3) Stream cipher setup

- Test Patterns
- TRNG
- Key Management and Storing
- Scan chain
- IV
- IDCODE
- BYP
- IR
- TAP controller

Original Circuit

- Stream Cipher
  - IV
  - Key
  - Keystream sout
  - Keystream sisn

Off-Chip Encryption
On-Chip Decryption
On-Chip Encryption
Off-Chip Decryption
INITIALIZATION PHASE

Initialization phase finished => Encryption phase

Test Patterns

TRNG

IV

Key

Keystream_so

Keystream_si

Scan chain

Original Circuit

Key Management and Storing

S_in

S_out

IR

TAP controller

Off-Chip Encryption

On-Chip Decryption

On-Chip Encryption

Off-Chip Decryption

Test Responses

TDI

TCK

TMS

TDI
**ENCRYPTION PHASE**

- User sends *GETIV* instruction
  
  $\Rightarrow$ Shift the content of the IV register out the circuit
**ENCRYPTION PHASE**

- User can encrypt and decrypt test data with the obtained IV and the shared secret key.
**TIME FOR THE INITIALIZATION PROCESS**

- $T_{TRNG\_init}$ to initialize the TRNG
- 80 clock cycles to shift the IV in the register
- 1 152 clock cycles for the stream cipher setup

<table>
<thead>
<tr>
<th>Original circuit</th>
<th>Triple-DES</th>
<th>Pipelined AES-128</th>
<th>Pipelined AES-256</th>
<th>RSA 1024</th>
<th>LEON3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test time*</td>
<td>687 101</td>
<td>1 944 877</td>
<td>4 559 845</td>
<td>39 405 239</td>
<td>11 612 051</td>
</tr>
<tr>
<td>(clock cycles)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Test time overhead**

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Block-based solution (%)</td>
<td>+0.31</td>
<td>+0.81</td>
<td>+0.006</td>
<td>+0.33</td>
<td>+0.004</td>
</tr>
<tr>
<td>Stream-based solution (%)**</td>
<td>+0.18</td>
<td>+0.06</td>
<td>+0.03</td>
<td>+0.003</td>
<td>+0.01</td>
</tr>
</tbody>
</table>

*: Test time considered for a fault coverage of 100%, except for LEON3 where it reaches 70%
**: test time overhead without the initialization of the TRNG
SUMMARY

1) Scan chain encryption

2) State-of-the-art based on test communication encryption

3) Implementation with block cipher

4) Implementation with stream cipher

5) Conclusion
## COMPARISON BETWEEN BOTH SOLUTIONS

<table>
<thead>
<tr>
<th></th>
<th>Block cipher-based solution (PRESENT)</th>
<th>Stream cipher-based solution (TRIVIUM)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Security</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Scan attacks</td>
<td>Protected</td>
<td>Protected (two times pad not possible)</td>
</tr>
<tr>
<td>- Malicious core</td>
<td>Protected</td>
<td>Protected</td>
</tr>
<tr>
<td><strong>Cost</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Area</td>
<td>10 658.96 µm²</td>
<td>5 408.52 µm² (+ 31 200 µm² for TRNG)</td>
</tr>
<tr>
<td>- Test time</td>
<td>Depends on the scan length (multiple or not of the block size)</td>
<td>Clock cycles required for the initialization phase</td>
</tr>
<tr>
<td><strong>Integration</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Diagnosis &amp; debug</td>
<td>Still possible in-field</td>
<td></td>
</tr>
<tr>
<td>- Key management</td>
<td>Re-use key management already implemented</td>
<td></td>
</tr>
<tr>
<td>- Integration in test</td>
<td>Possible issue with the padding of test data</td>
<td>No issue</td>
</tr>
<tr>
<td>daisy-chain</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ACKNOWLEDGEMENTS

- FUI#20 TEEVA Project

- Partners