

7th **INTERNATIONAL CONFERENCE ON**
PUBLIC KEY INFRASTRUCTURE AND
ITS APPLICATIONS (PKIA 2026)
SEPTEMBER 10-11th, 2026

A NIST SP 800-232 Compliant Validation Suite
for Unified Ascon Test Vector Generator

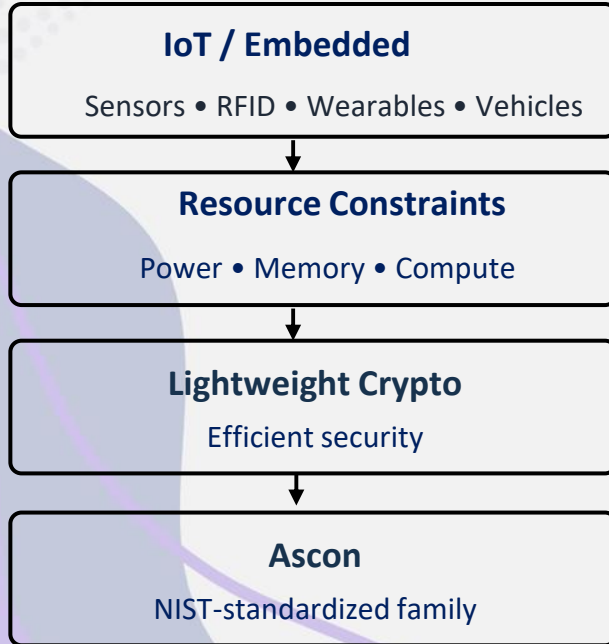
Umadevi Chikkadi · Rahul Kumar · Abey Jacob

Centre for Development of Advanced Computing (C-DAC), Bangalore, India

Agenda

- **Motivation: Lightweight Cryptography & Ascon**
- **Background: The Ascon Algorithm Family**
- **The Problem: A Missing Validation Infrastructure**
- **Proposed System: Unified Test Vector Generator**
- **Test Vector Generation Methodology**
- **Sample Vectors & Validation Coverage**

Motivation: Secure Cryptography for Resource-Constrained Devices



- NIST standard SP 800-232 specifies the list of algorithms:
 - Ascon-AEAD128 for authenticated encryption,
 - Ascon-Hash256 for fixed-length hashing
 - Ascon-XOF128 as an extendable-output hash function,
 - Ascon-CXOF128 as a customizable extendable-output hash function

How Ascon Works: Ascon Components

Ascon is based on sponge construction for all Ascon algorithms and uses 320-bit internal state value that is divided into 5 64-bit words consisting of rate 'r' and capacity 'c'. So, $r+c$ = internal state.

Ascon uses SPN(Substitution-Permutation-Network) consisting of 3 layers:

- **Constant-Addition Layer (p_C):** Adds round constants to disrupt symmetries.
- **Substitution Layer (p_S):** Uses a 5-bit S-Box to provide non-linearity (the "scrambling" core).
- **Linear Diffusion Layer (p_L):** Uses bitwise rotations to spread changes across the entire 320-bit state.

Together Permutation is given by: $p = p_C(p_S(p_L))$

— linear diffusion $\rightarrow$ 5-bit S-box $\rightarrow$ constant addition

Ascon-p[12]

12 rounds — initialization, finalization, all Hash / XOF / CXOF operations

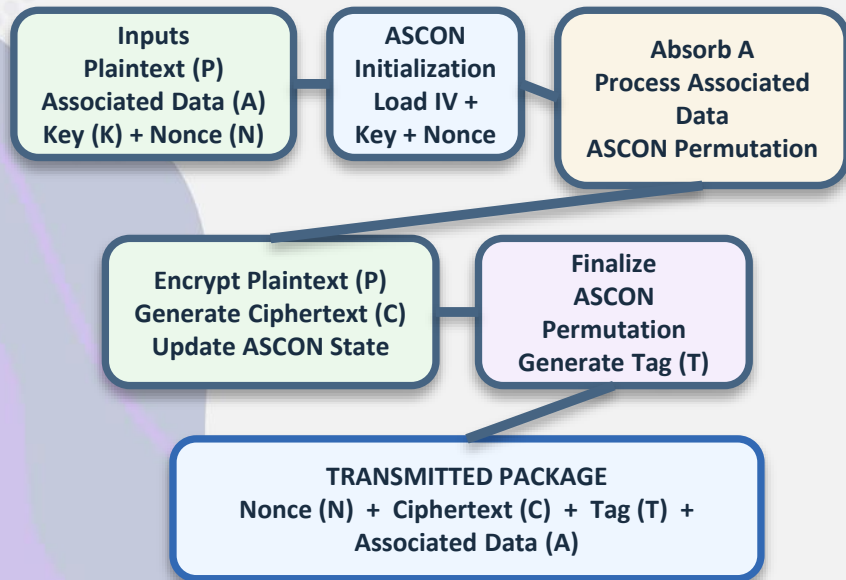
Ascon-p[8]

8 rounds — per-block associated-data / plaintext / ciphertext processing in AEAD128

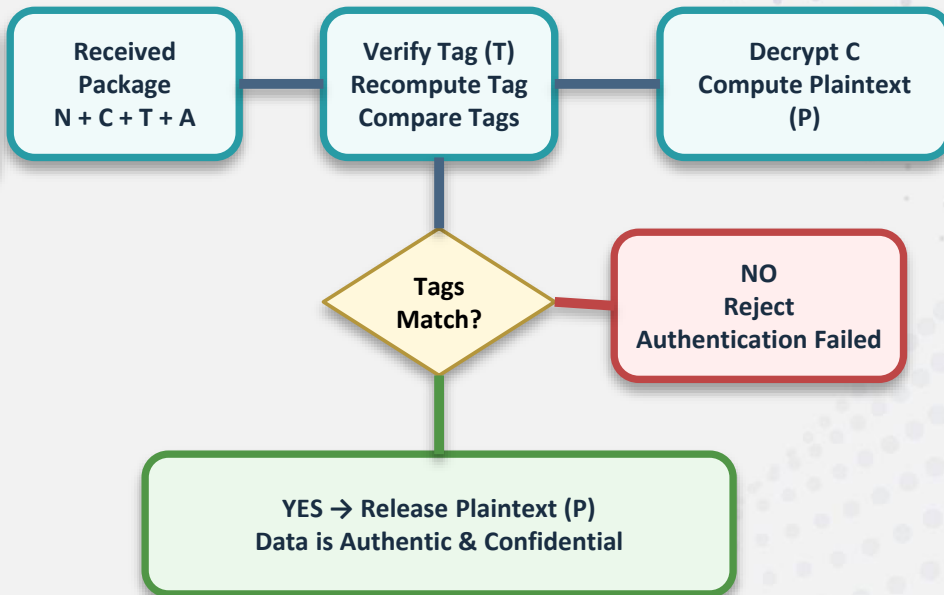
Ascon Variants	Rate	Capacity	State
Ascon_AEAD	128	192	320
Ascon_HASH	64	256	320
Ascon_XOF	64	256	320
Ascon CXOF	64	256	320

How Ascon Works: Ascon-AEAD

AUTHENTICATED ENCRYPTION



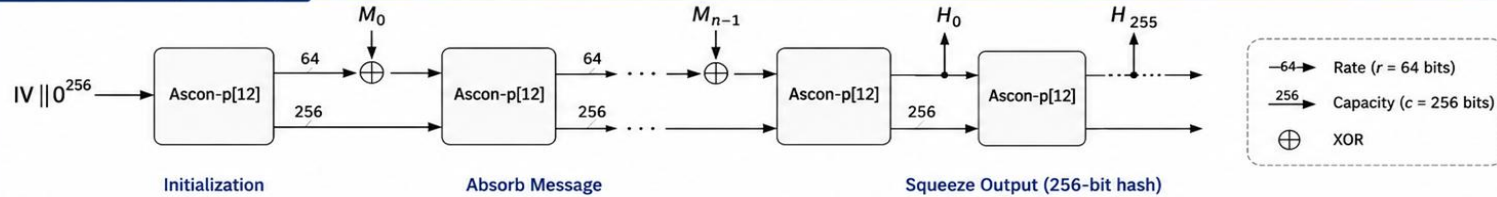
AUTHENTICATED DECRYPTION



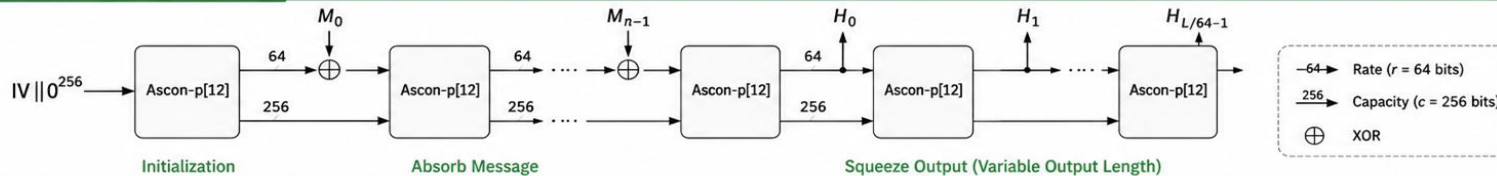
ASCON AEAD provides confidentiality of plaintext and authentication/integrity of both plaintext and associated data.

How Ascon Works: Ascon Hash256 Ascon-XOF and Ascon-CXOF

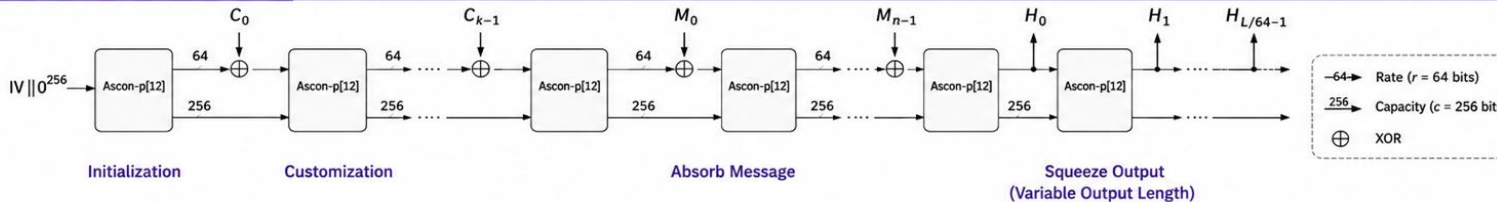
ASCON – Hash (Hash256)



ASCON – XOF (XOF128)



ASCON – CXOF (CXOF128)



Hash = Fixed Output | XOF = Variable Output | CXOF = Variable Output + Customization



<https://pkiindia.in>



Social Media
/pkiindia

Validation Test Categories Adopted from Standards

ISO/IEC 18367:2016(E) Validation Test for Hash

ISO/IEC 18367:2016(E) Standard specifies the following validation tests for bit and byte oriented modes of operations:

- Short Message Bit oriented mode
- Short Message Byte oriented mode
- Long Message Bit oriented mode
- Long Message Byte oriented mode
- Monte test in Byte oriented mode

ASCON Validation System

For the ASCON hash functions, the validation test includes:

- The Short Messages Test
- The Long Messages Test and
- The Pseudo Randomly Generated Messages (Monte Carlo) Test.

For the ASCON XOF and ASCON CXOF, the validation test includes:

- The Short Messages Test
- The Long Messages Test
- The Pseudo Randomly Generated Messages (Monte Carlo) Test; and
- The Variable-Length Output Test.

Short & Long Message Tests: Coverage by Design

- **Short Message Bit oriented mode**

Input Message 'Msg' will be in length 0, 1, 2 ... to 'r' bits that is 0 to $r + 1$ message. So there are $r+1$ messages in the request file REQ including 0 length.

- **Short Message Byte oriented mode**

Input message length will be in multiple of byte length 0, 8, 16, 24, 32 ... to 'r' that is 0 to $r/8 + 1$ messages. So there will be total of $r/8 + 1$ messages in the request file REQ including 0 length.

- **Long Message Bit oriented mode**

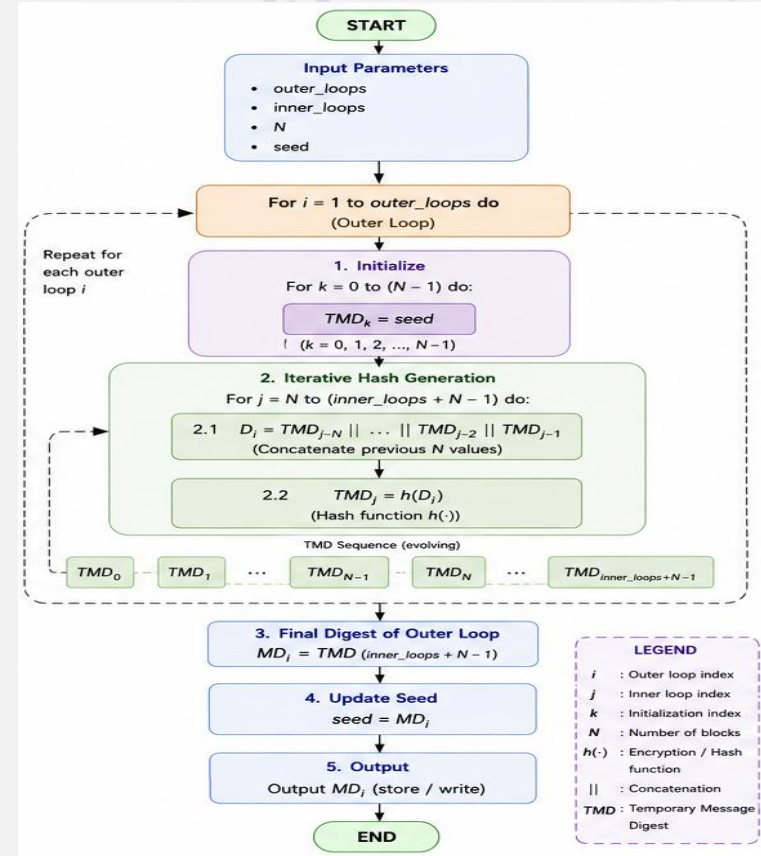
Input message length will be in sequence of 0 to $(r + 99*i)$ message length. Where, 'i' take values from 0 to 'r'. So there will be total of $r+1$ messages in the request file REQ including 0 length.

- **Long Message Byte oriented mode**

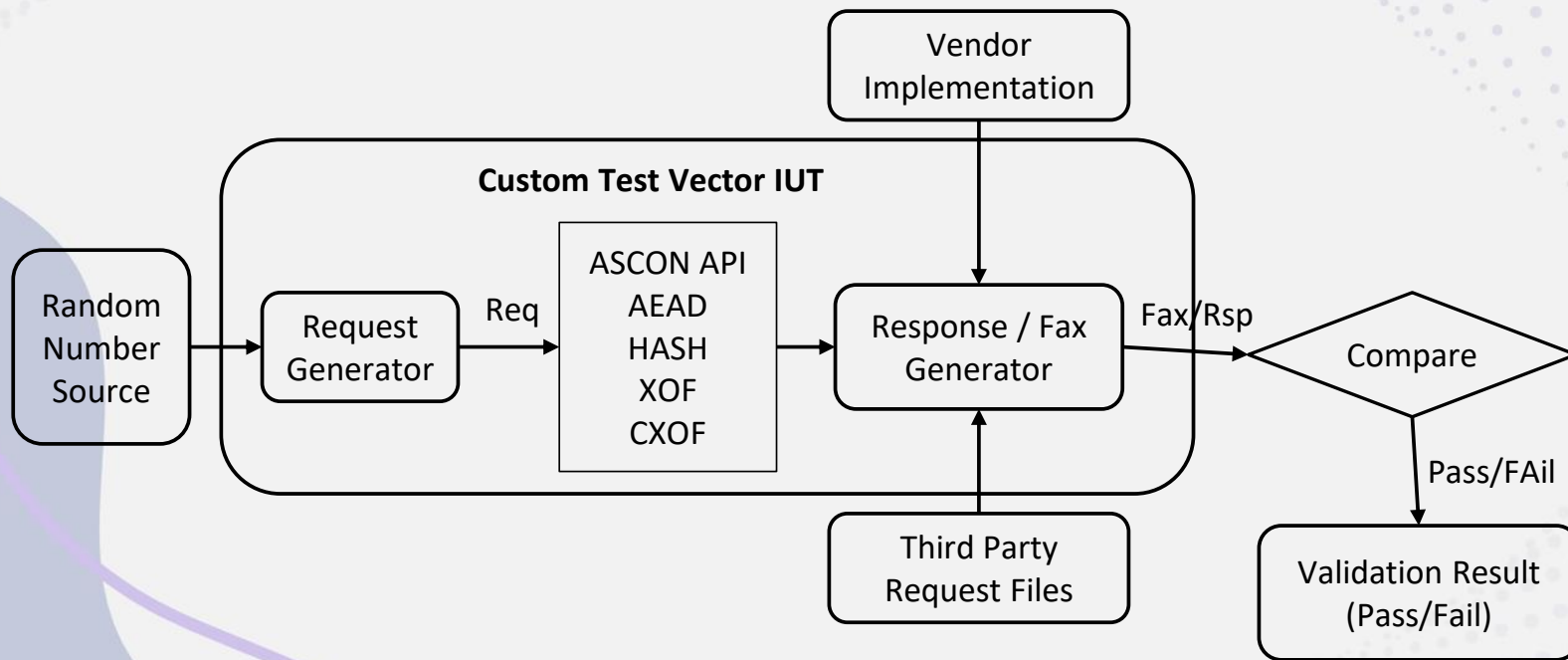
Message length will be in sequence of 0 to $(r + 8*99*i)$ message length. Where, 'i' takes values from 0 to $r/8$. So there will be total of $r/8 + 1$ messages in the request file REQ including 0 length.

The Monte Carlo Test

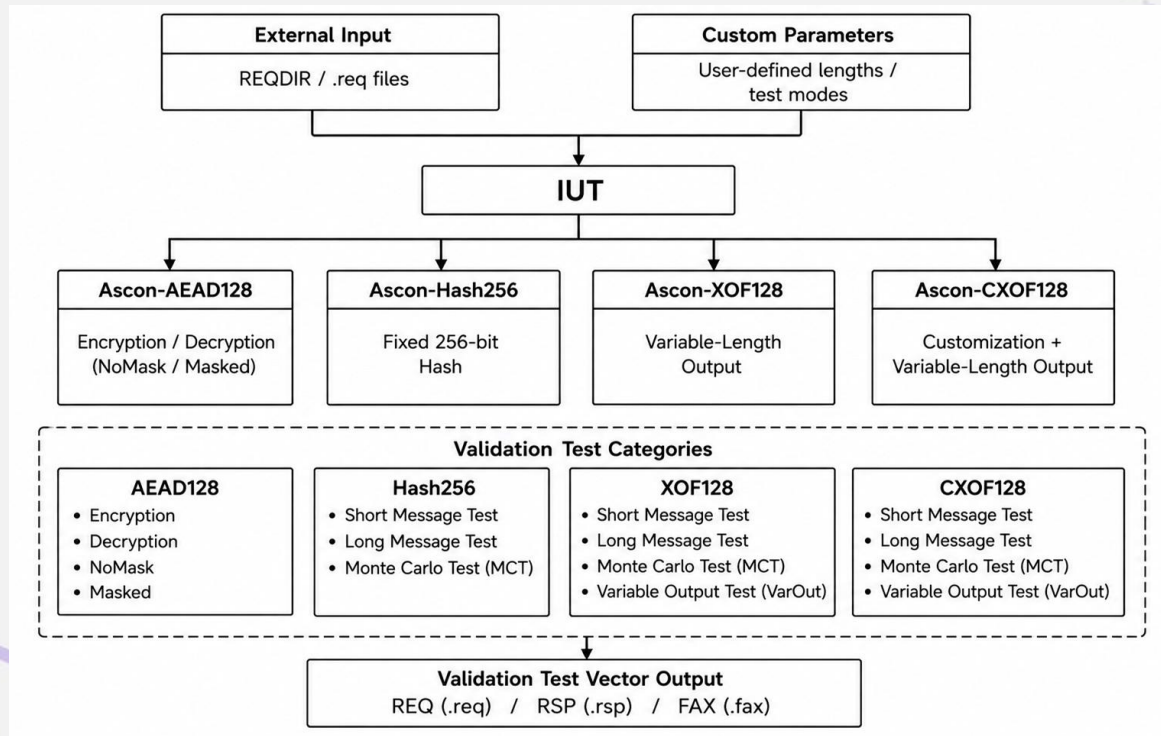
- **Why it matters:** Catches subtle, deterministic bugs — padding, byte-ordering, state handling — that a handful of isolated calls could miss.
- MCT alone runs 100,000 chained hash calls per file — strong amplification of subtle bugs
- Sliding window $N = 3$, 100 outer loops $\times$ 1,000 inner iterations
- Each inner step feeds the last 3 outputs back in as the next input — a tiny bug compounds and diverges visibly



Custom Test Vector Generation Framework and Process



Proposed System: Unified Test Vector Generator

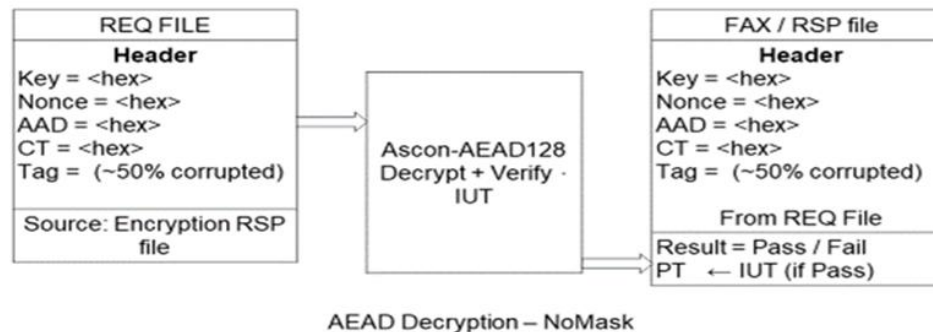
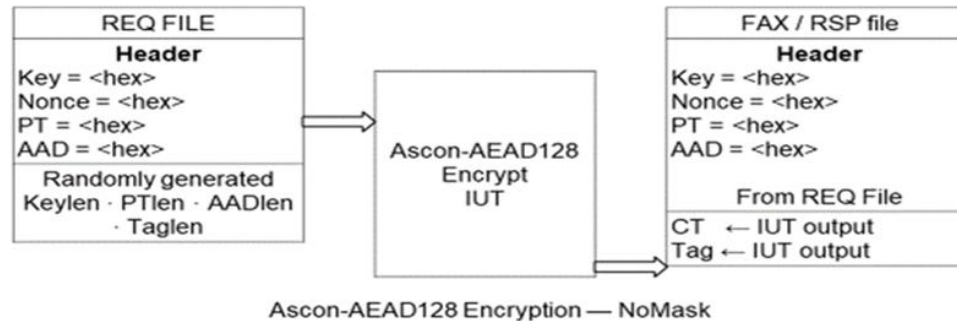


Validation Categories Supported by the Proposed Framework

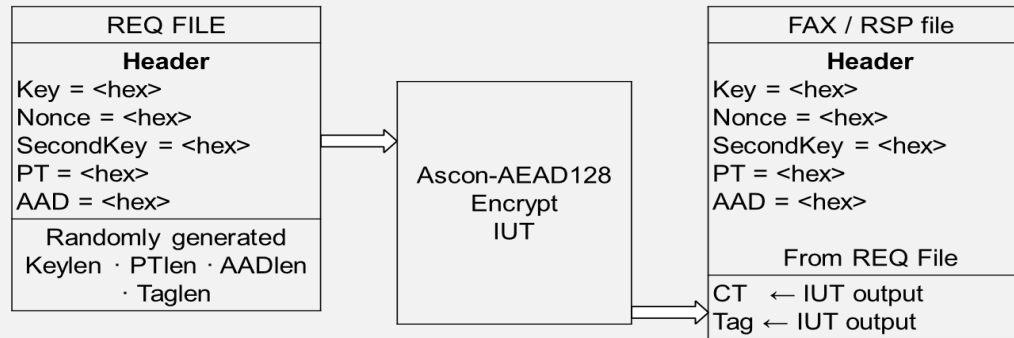
Algorithm	Categories Covered	Configuration Parameters
Ascon-AEAD128	Encryption, Decryption	Masking: NoMask / Masked
Ascon-Hash256	Short Message, Long Message, MCT	ShortMsg: 0–64 bits LongMsg: 856–6400 bits (byte), 163–6463 bits (bit) Output: Fixed 256 bits
Ascon-XOF128	Short Message, Long Message, MCT, Variable Output	ShortMsg: 0–64 bits LongMsg: 856–6400 bits (byte), 163–6463 bits (bit) VarOut Msg: 0–65536 bits VarOut Output: 8–65536 bits (byte), 1–65536 bits (bit)
Ascon-CXOF128	Short Message, Long Message, MCT, Variable Output	ShortMsg: 0–64 bits LongMsg: 856–6400 bits (byte), 163–6463 bits (bit) VarOut Msg: 0–65536 bits Output: 8–65536 bits (byte), 1–65536 bits (bit) Customization: 0–2048 bits

Short/Long Message vectors are cross-checked against independently derived NIST ACVP server vectors.

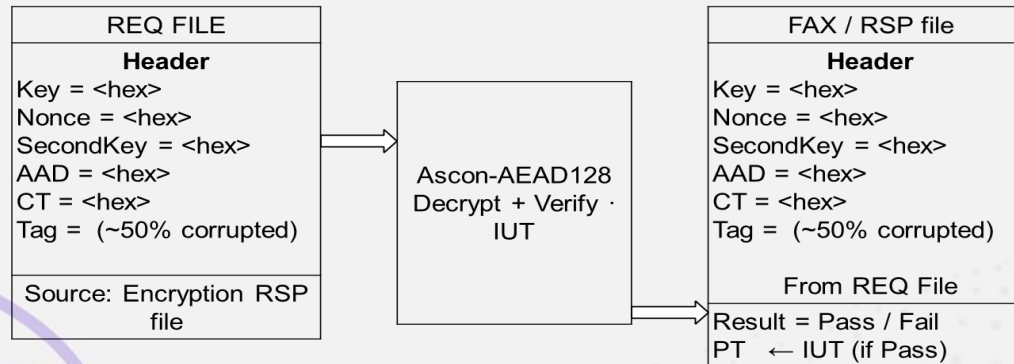
Test Vector Generation Framework for Ascon-AEAD128-NoMasked



Test Vector Generation Framework for Ascon-AEAD128-Masked

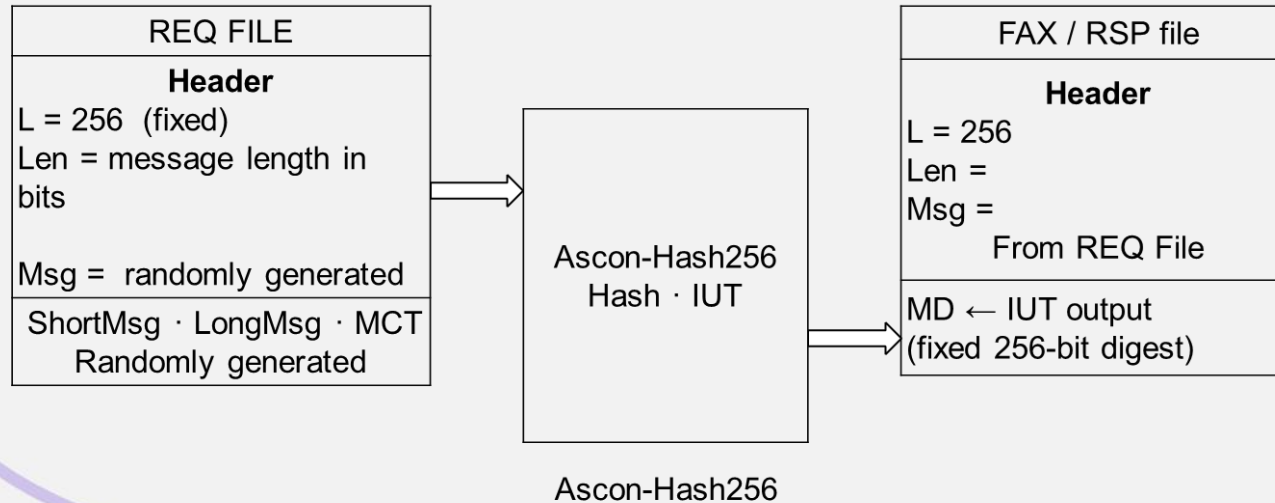


Ascon-AEAD128 Encryption — Masked

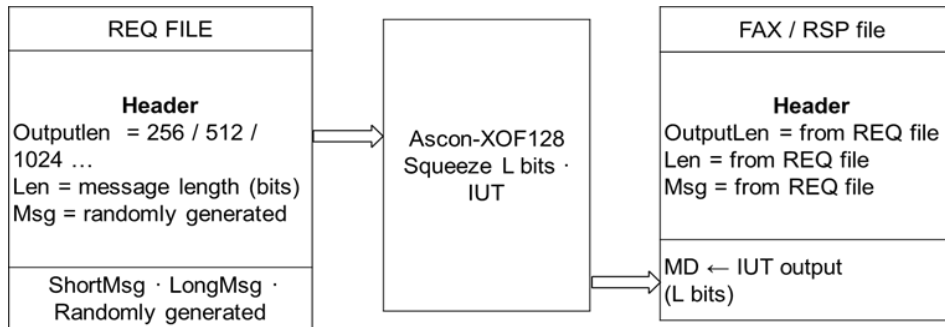


AEAD Decryption – Masked

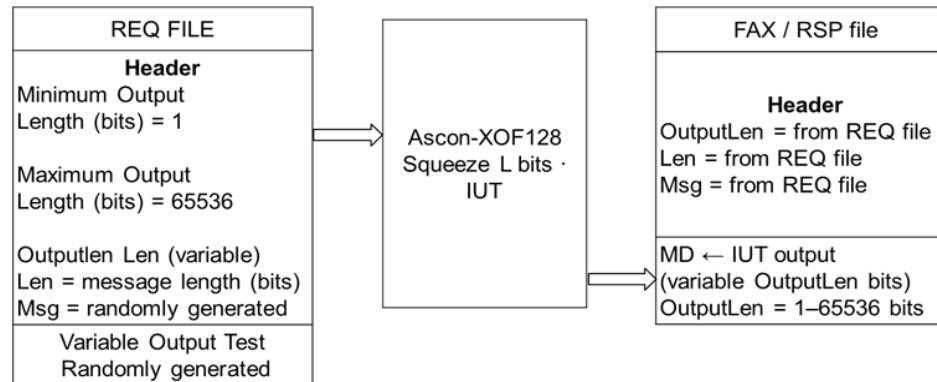
Test Vector Generation Framework for ASCON-Hash256



Test Vector Generation Framework for ASCON- XOF128

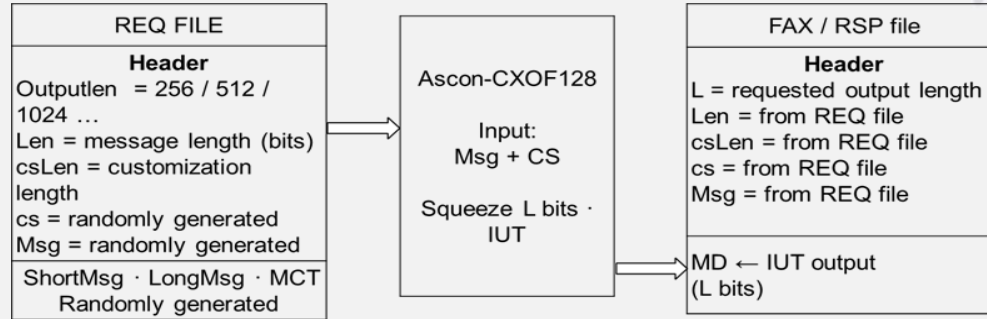


Ascon-XOF128 — Short/Long Message Test

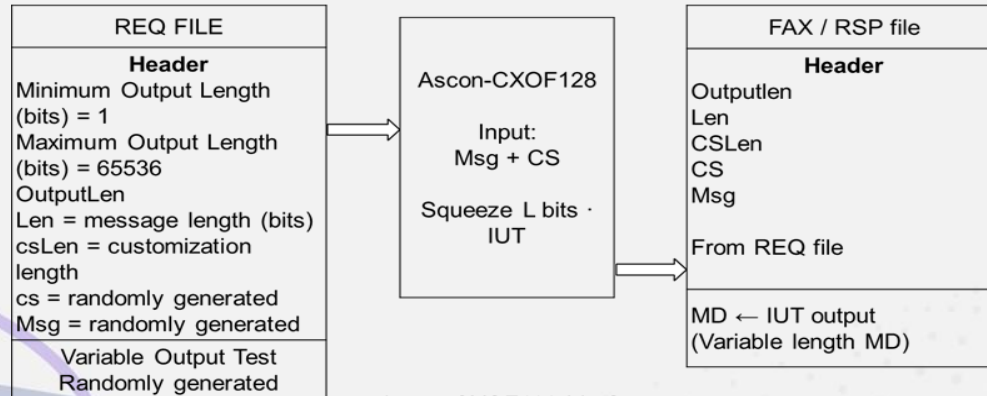


Ascon-XOF128 VarOut

Test Vector Generation Framework for ASCON-CXOF128



Ascon-CXOF128 — Short/Long Message Test



Ascon-CXOF128 VarOut

Directory Structure

```
rahul@HP:~/ascon_custom_test_vectors$ tree
.
├── ascon_aead
│   ├── testfile_req
│   │   └── ASCON128
│   │       ├── ASCON128Dec
│   │       │   ├── AEAD_Decrypt_Masked.req
│   │       │   ├── AEAD_Decrypt_Masked.sam
│   │       │   ├── AEAD_Decrypt_NoMask.req
│   │       │   └── AEAD_Decrypt_NoMask.sam
│   │       └── ASCON128Enc
│   │           ├── AEAD_Encrypt_Masked.req
│   │           └── AEAD_Encrypt_NoMask.req
│   └── testfile_rsp
│       └── ASCON128
│           ├── ASCON128Dec
│           │   ├── AEAD_Decrypt_Masked.rsp
│           │   └── AEAD_Decrypt_NoMask.rsp
│           └── ASCON128Enc
│               ├── AEAD_Encrypt_Masked.rsp
│               └── AEAD_Encrypt_NoMask.rsp
```

```
ascon_hash
├── testfile_req
│   └── ASCON128
│       ├── Ascon_CX0F
│       │   ├── Ascon-CX0F128_LongMsg_Bit.req
│       │   ├── Ascon-CX0F128_LongMsg_Byte.req
│       │   ├── Ascon-CX0F128_MCT_Bit.req
│       │   ├── Ascon-CX0F128_MCT_Byte.req
│       │   ├── Ascon-CX0F128_ShortMsg_Bit.req
│       │   ├── Ascon-CX0F128_ShortMsg_Byte.req
│       │   ├── Ascon-CX0F128_VarOut_Bit.req
│       │   └── Ascon-CX0F128_VarOut_Byte.req
│       ├── Ascon_Hash
│       │   ├── Ascon-Hash256_LongMsg_Bit.req
│       │   ├── Ascon-Hash256_LongMsg_Byte.req
│       │   ├── Ascon-Hash256_MCT_Bit.req
│       │   ├── Ascon-Hash256_MCT_Byte.req
│       │   ├── Ascon-Hash256_ShortMsg_Bit.req
│       │   └── Ascon-Hash256_ShortMsg_Byte.req
│       └── Ascon_X0F
│           ├── Ascon-X0F128_LongMsg_Bit.req
│           ├── Ascon-X0F128_LongMsg_Byte.req
│           ├── Ascon-X0F128_MCT_Bit.req
│           ├── Ascon-X0F128_MCT_Byte.req
│           ├── Ascon-X0F128_ShortMsg_Bit.req
│           ├── Ascon-X0F128_ShortMsg_Byte.req
│           ├── Ascon-X0F128_VarOut_Bit.req
│           └── Ascon-X0F128_VarOut_Byte.req
```

```
testfile_rsp
├── ASCON128
│   ├── Ascon_CX0F
│   │   ├── Ascon-CX0F128_LongMsg_Bit.rsp
│   │   ├── Ascon-CX0F128_LongMsg_Byte.rsp
│   │   ├── Ascon-CX0F128_MCT_Bit.rsp
│   │   ├── Ascon-CX0F128_MCT_Byte.rsp
│   │   ├── Ascon-CX0F128_ShortMsg_Bit.rsp
│   │   ├── Ascon-CX0F128_ShortMsg_Byte.rsp
│   │   ├── Ascon-CX0F128_VarOut_Bit.rsp
│   │   └── Ascon-CX0F128_VarOut_Byte.rsp
│   ├── Ascon_Hash
│   │   ├── Ascon-Hash256_LongMsg_Bit.rsp
│   │   ├── Ascon-Hash256_LongMsg_Byte.rsp
│   │   ├── Ascon-Hash256_MCT_Bit.rsp
│   │   ├── Ascon-Hash256_MCT_Byte.rsp
│   │   ├── Ascon-Hash256_ShortMsg_Bit.rsp
│   │   └── Ascon-Hash256_ShortMsg_Byte.rsp
│   └── Ascon_X0F
│       ├── Ascon-X0F128_LongMsg_Bit.rsp
│       ├── Ascon-X0F128_LongMsg_Byte.rsp
│       ├── Ascon-X0F128_MCT_Bit.rsp
│       ├── Ascon-X0F128_MCT_Byte.rsp
│       ├── Ascon-X0F128_ShortMsg_Bit.rsp
│       ├── Ascon-X0F128_ShortMsg_Byte.rsp
│       ├── Ascon-X0F128_VarOut_Bit.rsp
│       └── Ascon-X0F128_VarOut_Byte.rsp
```

21 directories, 54 files

Sample Generated Vectors

```
rahul@HP: ~/ascon_custom_test_vectors/ascon_aead/testfile_rsp/AS...  
# CDAC Algorithm Validation Suite 1.0  
# Ascon-AEAD128 Encryption Test Vectors without Nonce Masking  
# Generated on Thu Sep 03 18:28:22 2026  
  
[Keylen = 128]  
[Noncelen = 128]  
[PTlen = 0]  
[AADlen = 0]  
[Taglen = 128]  
  
Count = 0  
Key = 5fad0ee8131b4b91da2c9d78112a7ca8  
Nonce = ec8da51d6cf734deeb124459161ed230  
PT =  
AAD =  
CT =  
Tag = fce2013209cce3f0747155007c770a76  
  
Count = 1  
Key = 361798f5f1c60a43ef99dd1134aa096f  
Nonce = 5a84f6151def32ea0870452b423eae3b  
PT =  
AAD =  
CT =  
Tag = 37b20eef27b8464716cd55c890be7df5
```

Sample RSP
for AEAD
ENCRYPT
and
DECRYPT

```
rahul@HP: ~/ascon_custom_test_vectors/ascon_aead/testfile_rsp/ASCON1...  
# CDAC Algorithm Validation Suite 1.0  
# Ascon-AEAD128 Decryption Response Vectors without Nonce Masking  
# Generated on Thu Sep 03 18:28:23 2026  
  
[Keylen = 128]  
[Noncelen = 128]  
[PTlen = 0]  
[AADlen = 0]  
[Taglen = 128]  
  
Count = 0  
Key = bfdde835f037a07c61bc643d7237c8b5  
Nonce = 46035fb9f52bc43e9b0f9af27ed4ff13  
AAD =  
CT =  
Tag = a9016ee3aaef204a749c26c8cf99ed1e  
Result = Fail  
Reason = modified tag  
  
Count = 1  
Key = 2e027a35f3966c69be5ef31c337db704  
Nonce = 5c5300274ce000e572540a4cceef0e2e8  
AAD =  
CT =  
Tag = 5e1c8f1a4fa629e30afa923106b7b5fa  
Result = Pass  
Reason = valid input - ciphertext should decrypt successfully  
PT =
```

Sample Generated Vectors

```
rahul@HP: ~/ascon_custom_test_vectors/ascon_hash/testfile_rsp/ASCON...  
# CDAC Algorithm Validation Suite 1.0  
# Ascon-CXOF128  
# Monte Carlo Test (Bit-oriented)  
# Generated on 2026-09-04 09:59:23  
  
[L = 256]  
[CSLen = 0]  
[CS = ]  
  
Count = 0  
Seed = 4e5575059ab1420a8da9ddbf084549c5c5eb061380fc867c5c49a8cf0eccc06f  
MD = dc1157db69d1713db8d07b3039127285f3a3ab8bad27f2df3a949b726c66a1fb  
  
Count = 1  
Seed = dc1157db69d1713db8d07b3039127285f3a3ab8bad27f2df3a949b726c66a1fb  
MD = be03ad0f08e9029f804ea86aa400f66c3680de160bcf7deb92f6f954c5739c2c  
  
Count = 2  
Seed = be03ad0f08e9029f804ea86aa400f66c3680de160bcf7deb92f6f954c5739c2c  
MD = 735742b2e7045e298f1b61dbb22cf23c4792f14e5eb45ec67bc142993f4fe2ca  
  
Count = 3  
Seed = 735742b2e7045e298f1b61dbb22cf23c4792f14e5eb45ec67bc142993f4fe2ca  
MD = 476f71a3866f59d45d2789f822955f5ba3ee7eeddf1ad42861a5e188373a6c5f  
  
Count = 4  
Seed = 476f71a3866f59d45d2789f822955f5ba3ee7eeddf1ad42861a5e188373a6c5f  
MD = f158a69e81c6db0a53ea58e44a7f4e98c24732f748b0d17523c79bcee634e6c8  
  
Count = 5  
Seed = f158a69e81c6db0a53ea58e44a7f4e98c24732f748b0d17523c79bcee634e6c8  
MD = 91e5c7b0d5a4b385609fbe4c93eec62234af1809670028dad4a4535f82883335  
  
Count = 6  
Seed = 91e5c7b0d5a4b385609fbe4c93eec62234af1809670028dad4a4535f82883335
```

Sample RSP for
Ascon CXOF MCT

Sample Generated Vectors

```
rahu@HP: ~/ascon_custom_test_vectors/ascon_hash/testfile_rsp/ASCON128/Ascon_...  
# CDAC Algorithm Validation Suite 1.0  
# Ascon-XOF128  
# Variable Output Length Test (VarOut, Bit-oriented)  
# Generated on 2026-09-04 09:55:48  
  
[Tested for Output of bit-oriented messages]  
[Minimum Output Length (bits) = 1]  
[Maximum Output Length (bits) = 65536]  
  
[Input Length = 0]  
  
Count = 0  
Outputlen = 1  
Len = 0  
Msg =  
CSLen = 01  
  
Count = 1  
Outputlen = 512  
Len = 0  
Msg =  
MD = 473d5e6164f58b39dfd84aacdb8ae42ec2d91fed33388ee0d960d9b3993295c6ad77  
855a5d3b13fe6ad9e6098988373af7d0956d05a8f1665d2c67d1a3ad10ff
```

**Sample RSP
for Ascon
XOF &
CXOF
VarOut**

```
rahu@HP: ~/ascon_custom_test_vectors/ascon_hash/testfile_rsp/ASCON128/Ascon_...  
# CDAC Algorithm Validation Suite 1.0  
# Ascon-CXOF128  
# Variable Output Length Test (VarOut, Bit-oriented)  
# Generated on 2026-09-04 09:59:25  
  
[Tested for Output of bit-oriented messages]  
[Minimum Output Length (bits) = 1]  
[Maximum Output Length (bits) = 65536]  
  
[Input Length = 0]  
  
Count = 0  
Outputlen = 1  
Len = 0  
Msg =  
CSLen = 0  
CS =  
MD = 01  
  
Count = 1  
Outputlen = 512  
Len = 0  
Msg =  
CSLen = 32  
CS = 1ea33e0a  
MD = 98076cfa0a0820badf919e7fee997a37790a26d7f783c58d24d761ea06e1407d1e6b  
d6dacc74301f8244523733811593a0cedc38b795c7876ebfafd95c7387bf
```

References

- NIST SP 800-232, “Ascon-Based Lightweight Cryptography Standards for Constrained Devices,” 2025.
- ISO/IEC 18367:2016(E), “Information technology — Security techniques — Cryptographic algorithms and security mechanisms conformance testing.”
- NIST, FIPS 140-3: Security Requirements for Cryptographic Modules, National Institute of Standards and Technology, Gaithersburg, MD, 2019.
- NIST ACVP-Server / generated validation vectors.
- NIST Cryptographic Module Validation Program (CMVP).
- Dobraunig et al., “Ascon v1.2: Lightweight Authenticated Encryption and Hashing,” J. Cryptol., 2021.
- U. Devi and A. Jacob, “Cryptographic Validation of Lightweight Block ciphers and Hash Functions,” in 2023 IEEE International Conference on Public Key Infrastructure and its Applications (PKIA), Bangalore, India
- <https://csrc.nist.gov/projects/lightweight-cryptography>

THANK YOU