KIASU

Jérémy Jean - Ivica Nikolić - Thomas Peyrin

NTU - Singapore

DIAC 2014

Santa Barbara, USA - August 23, 2014

http://www1.spms.ntu.edu.sg/~syllab/KIASU

NANYANG TECHNOLOGICAL UNIVERSITY

Summary

- first and only adhoc tweakable AES-128 ...
- ... which allows us to provide 2^{128} guarantee for both integrity and forgery - no birthday security !
- extremely fast in software, on par with OCB3 for long messages
- fast for short messages - minimal overhead as no initialization is needed
- quite small in hardware
- parallelizable
- very simple - almost direct plug-in of AES-128 (reuse existing security analysis and implementations), backward compatible with AES-128
- we provide a nonce-misuse resistant mode if needed

Outline

KIASU≠, KIASU= and KIASU-BC

We have two operating modes KIASU≠ and KIASU= , both built upon the same tweakable block cipher named KIASU-BC .

Operating modes:

- KIASU≠ is for nonce-respecting (based on OCB3)
- KIASU= is for nonce-misuse resistance (based on COPA)
- both modes are parallelizable

The tweakable block cipher KIASU-BC :

- message of $n = 128$ bits
- key of $k = 128$ bits
- tweak of $t = 64$ bits

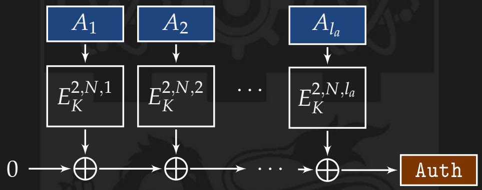
Outline

1. **Description of KIASU**
   - ▷ Operating mode(s)
   - ▷ The tweakable block cipher KIASU-BC
2. **Security**
3. **Performances and Features**
4. **Conclusion**

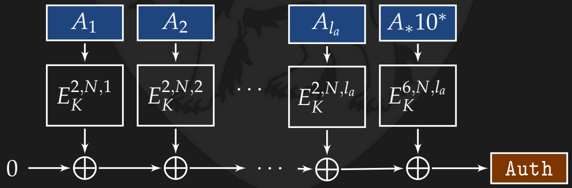
nonce-respecting mode: KIASU≠

KIASU≠ is based on OCB3

For Associated Data (full block):



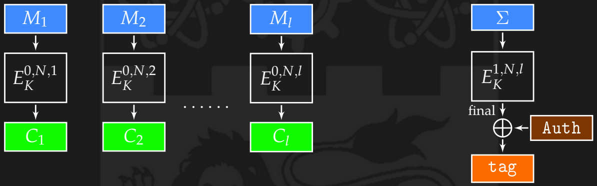
For Associated Data (partial block):



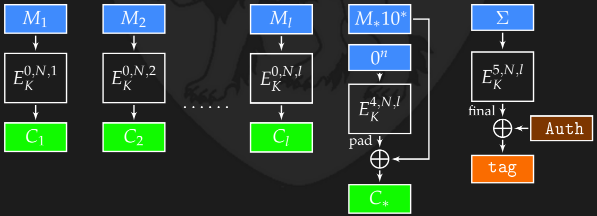
nonce-respecting mode: KIASU≠

KIASU≠ is based on OCB3

For Plaintext (full block):



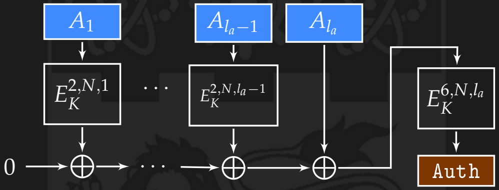
For Plaintext (partial block):



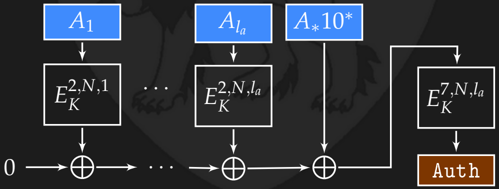
nonce-misuse resistant mode: KIASU≠

KIASU≠ is based on COPA

For Associated Data (full block):



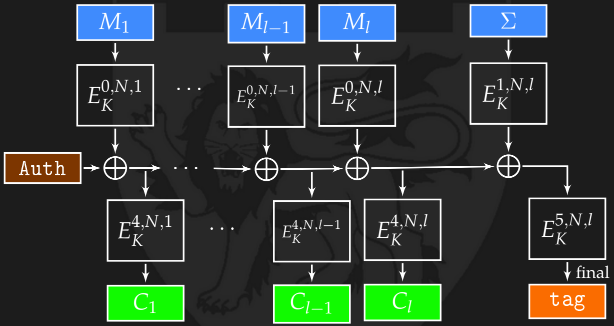
For Associated Data (partial block):



nonce-misuse resistant mode: KIASU≠

KIASU≠ is based on COPA

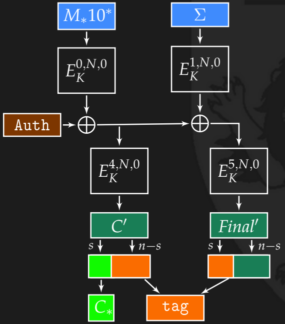
For Plaintext (full block):



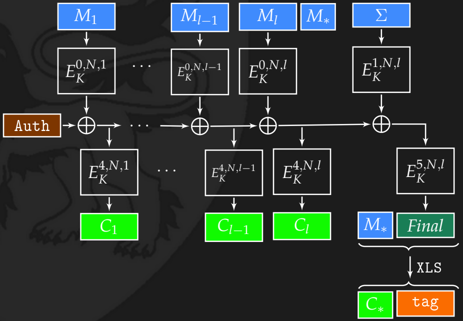
nonce-misuse resistant mode: KIASU≠

KIASU≠ is based on COPA

For Plaintext (single partial block):



For Plaintext (partial block):

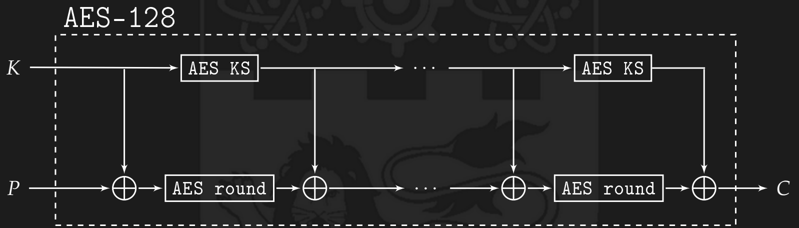


Outline

1. **Description of KIASU**
   - Operating mode(s)
   - The tweakable block cipher KIASU-BC
2. **Security**
3. **Performances and Features**
4. **Conclusion**

The tweakable block cipher KIASU-BC

KIASU-BC is **exactly** the AES-128 cipher, but with a fixed 64-bit tweak value T XORed to each subkey (on the two first rows).

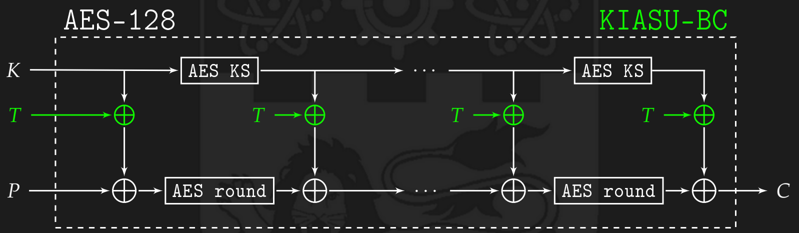


$$T = \begin{array}{|c|c|c|c|} \hline T_0 & T_2 & T_4 & T_6 \\ \hline T_1 & T_3 & T_5 & T_7 \\ \hline 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ \hline \end{array}$$

TWEAKEY framework (see next presentation and AsiaCrypt 2014)

The tweakable block cipher KIASU-BC

KIASU-BC is **exactly** the AES-128 cipher, but with a fixed 64-bit tweak value T XORed to each subkey (on the two first rows).



$T =$

T_0	T_2	T_4	T_6
T_1	T_3	T_5	T_7
0	0	0	0
0	0	0	0

TWEAKEY framework (see next presentation and AsiaCrypt 2014)

Outline

1. Description of KIASU
   - Operating mode(s)
   - The tweakable block cipher KIASU-BC
2. Security
3. Performances and Features
4. Conclusion

Security claims (in $\log_2$)

nonce-respecting user	Security (bits) KIASU≠	KIASU=
Confidentiality for the plaintext	128	64
Integrity for the plaintext	128	64
Integrity for the associated data	128	64

nonce-misuse user	Security (bits) KIASU≠	KIASU=
Confidentiality for the plaintext	none	64
Integrity for the plaintext	none	64
Integrity for the associated data	none	64

Conjectured security claims (in $\log_2$)

nonce-respecting user	Security (bits) KIASU≠	KIASU=
Confidentiality for the plaintext	128	128
Integrity for the plaintext	128	128
Integrity for the associated data	128	128

nonce-misuse user	Security (bits) KIASU≠	KIASU=
Confidentiality for the plaintext	none	64
Integrity for the plaintext	none	64
Integrity for the associated data	none	64

Security of KIASU-BC

The security of KIASU-BC is the same as AES-128 for a fixed tweak. The tricky part is to analyse what happens when the tweak varies.

If the key is fixed and one varies the tweak:

KIASU-BC's tweak schedule has been chosen such that it is itself a good key schedule.
Bad idea: adding a tweak on the entire 128-bit state, since trivial and very good related-tweak differential paths would exist.

If both the key and tweak vary:

KIASU-BC was designed such that no interesting interaction between the key schedule and the tweak schedule will exist. We put a special focus on attacks which are highly impacted by the key schedule:

- related-key related-tweak attacks
- meet-in-the-middle attacks

Security of KIASU-BC

related-key related-tweak attacks

We prove that no good related-key related-tweak attacks differential path exist for KIASU (even boomerang), with a computer-aided search tool.

rounds	active SBoxes	upper bound on probability	method used
1-2	0	2^{0}	trivial
3	1	2^{-6}	Matsui's
4	8	2^{-48}	Matsui's
5	≥ 14	2^{-84}	Matsui's
7	≥ 22	2^{-132}	ex. split (3R+4R)

Security proofs on operating modes

When the nonce is not reused, we ensure that
every call to KIASU-BC will have a distinct tweak input value

We can directly reuse the OCB3 and COPA operating modes security proofs.

- but we can ensure full 128-bit security
- the proofs are simpler (see ΘCB3 and OCB3 proofs)

Universal hash based tweakable block ciphers won't provide full 128-bit security (or with bad efficiency), due to the possibility of collisions between the inputs/outputs of the internal block cipher

Outline

1. Description of KIASU
   - Operating mode(s)
   - The tweakable block cipher KIASU-BC
2. Security
3. Performances and Features
4. Conclusion

Measuring authenticated encryption speed

One should consider several scenarios when measuring speed for AE:

- $K_\Delta N_\Delta$: when key and nonce are random (what SUPERCOP is currently measuring ?)
- $K_\Delta N_+$: when key is random, but nonce is counter
- $K_= N_\Delta$: when key is fixed, but nonce is random
- $K_= N_+$: when key is fixed, and nonce is counter
- $K_= N_=$: when both key and nonce are fixed (for nonce-misuse resistant schemes)

It would be great to measure all these 5 cases in SUPERCOP to get a better picture (probably $K_\Delta N_\Delta$ and $K_\Delta N_+$ are very similar)

When people present speed results, they should make clear in which of these 5 cases they made the measurements.

`KIASU` is rather neutral with regards to the first 4 cases (having $K_= N_\Delta$ or $K_= N_+$ makes no difference)

Software performances (in c/B) - case $K_\triangle N_\triangle$

both KIASU≠ and KIASU= can be parallelized

KIASU≠	**512B**	**1024B**	**4096B**	**65536B**
Intel Haswell	1.37	1.04	0.80	**0.72**
Intel Sandy Bridge	2.05	1.61	1.15	**0.99**
Intel Haswell (no AES-NI)	19.31	13.47	9.08	**7.71**

KIASU=	**512B**	**1024B**	**4096B**	**65536B**
Intel Haswell	2.32	1.88	1.59	**1.49**
Intel Sandy Bridge	3.79	3.13	2.55	**2.06**
Intel Haswell (no AES-NI)	26.77	20.91	16.61	**15.22**

Software performances (in c/B) - Fast on small messages

KIASU is fast for small messages, as it requires no initialization.

- sponge-like designs require strong initialization, AES-GCM-like designs usually prepare computation tables
- "simple IMIX" is a weighted average simulating sizes of typical IP packages:
  7 parts of 40B, 4 parts of 576B, 1 part of 1500B
- maximum transmission unit (MTU) for Ethernet is 1500 bytes

KIASU≠	40B	576B	1500B	IMIX
Intel Haswell	9.45	1.31	0.96	1.74
Intel Sandy Bridge	10.85	2.01	1.51	2.43

KIASU=	40B	576B	1500B	IMIX
Intel Haswell	25.03	2.30	2.30	3.86
Intel Sandy Bridge	31.53	3.54	3.64	5.50

Hardware performances

- easy to reuse existing tricks from AES-128 FPGA/ASIC implementations
- save implementation and area cost if co-implemented with AES-128
- being fast for small messages is very valuable, as small messages is a typical use-case of hardware applications

For FPGA (ongoing work):

- Marc Stöttinger and He Wei from NTU worked on a first (not yet optimized) round-based FPGA implementation of KIASU-BC
- 1989 slices (neither internal BRAM nor external RAM) for 1.08Gbit/s throughput on a Virtex-5 FPGA

For ASIC (ongoing work):

- we estimate that KIASU-BC can be implemented with 3000GE (reusing smallest know AES-128 implementation - 2400 GE)
- we estimate that one has to add an extra 1000 GE for implementing KIASU$\neq$, and 2000 GE for KIASU$=$

Others features

KIASU-BC is backward compatible with AES-128: simply set $T = 0$. This will save implementation overheads

KIASU will perform well on many platforms, even legacy ones, since it is very close to AES-128. This might not be true for candidates that perform multiplications in a big Galois field

tweakable block ciphers are very useful building blocks:

- block cipher, stream cipher
- parallel MAC
- parallel authenticated encryption: like OCB3 or COPA, but simpler design/proofs and much higher security bounds
- hash function: use the tweak input as block counter (HAIFA framework) or to perform randomized hashing
- tree hashing: use the tweak to encode the position in the tree
- PRNG, KDF, disk encryption

Outline

1. Description of KIASU
   - Operating mode(s)
   - The tweakable block cipher KIASU-BC
2. Security
3. Performances and Features
4. Conclusion

KIASU

KIASU-BC is the first AES-based ad-hoc tweakable block cipher

KIASU:

- ✓ faster than AES-GCM: extremely fast in software, especially for the message sizes that matter
- ✓ smaller than AES-GCM: good hardware profile
- ✓ more versatile than AES-GCM: good performances in any platform
- ✓ much higher security than AES-GCM: full 128-bit security
- ✓ much simpler than AES-GCM: simple design and proofs
- ✓ more features than AES-GCM: can easily switch to a nonce-misuse resistant mode
- ✓ parallelizable

Thank you !