Dual Microblaze Rekeying Processor for Group Key Management

Jose M. Granado-Criado, Miguel A. Vega-Rodríguez , Juan M. Sánchez-Pérez, Juan A. Gómez-Pulido

1. Introduction

Services like pay-per-view, IP television or satellite television needs a high computational effort to guarantee the user the access to the contracted services, which could be partially or totally the same for another user.

There is two techniques to make the key management:

- Basic technique
- Logical Key Hierarchy (LKH)

The number of encryptions perform for these two techniques is shown in the next table:

	Basic technique	LKH
User join	2	$2*\log_2 n$
User disjoin	$n - 1$	$2 * (\log_2 n - 1)$
Average	O(n)	O(log n)

2. Logical Key Hierarchy

LKH consists of dividing the group into hierarchical subgroups Every subgroup has a key (help-key) shared by all users of that subgroup. The root help-key represents the group key which is used to encrypt payload data. Every user stores several keys: its identifier key (Ki), the group key (Kg), and every help-key (Ki-j) among the group key and that user in the key tree.

2.1. Memory organization

Two memory structures are needed to manage the LKH:

- Group Key Memory (GKM): It stores all the key tree.
- Key State Memory (KST): For each help-key and the group key, KST stores a two bit state which: 00 (no children), 01 (some child in the right branch), 10, (some child in the left branch) or 11 (some child in both branches)

The following figures shows an eight users GKM.

Level	Addres	Key
0	0001	K_g
1	0010	K_{0-3}
	0011	K_{4-7}
2	0100	K_{0-1}
	0101	K_{2-3}
	0110	K_{4-5}
	0111	K_{6-7}

Level	Addres	Key
3	1000	K_0
	1001	K_1
	1010	K_2
	1011	K_3
	1100	K_4
	1101	K_5
	1110	K_6
	1111	K_7

2.2. Join and Disjoin

When a user join or disjoin to a group, several keys must be recalculated, encrypted and sent to the corresponding users.

The following pseudocode shows the operations make in a join operation:

Calculate a new user key
Send this key to the new user
Allocate the key in a sheet of the tree
While actual node is not the root node
  Go to the new user's father node
  Calculate a new key
  Encrypt this key with its left son key
  Send the encrypted help key to all users in its left branch
  Encrypt this key with its right son key
  Send the encrypted help key to all users in its right branch
End while

For a disjoin operation, the outgoing user key is erased from the tree and only the *while* loop is performed.

2.3. Key Generation

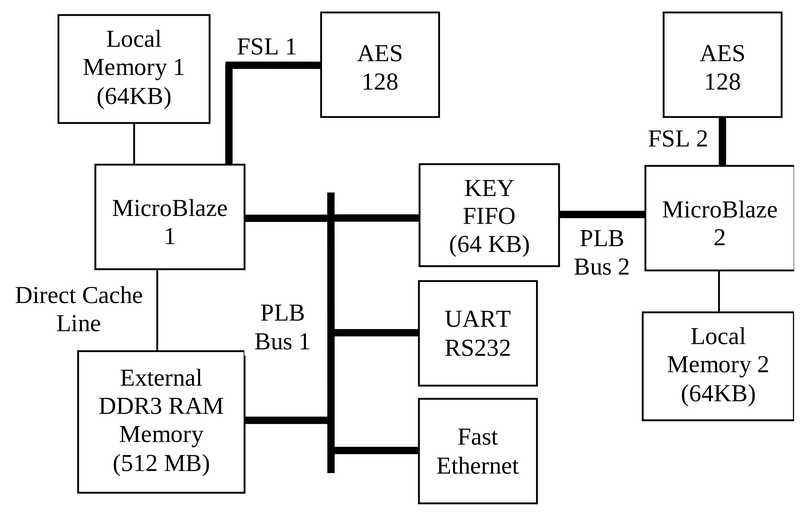
The ANSI X9.17 key generator is employed to generate new keys. The pseudocode of key generation is:

$I=E\_K_{gen}(D)$ *(initialization process)*
$K_i^{new} = E\_K_{gen}(I \text{ xor } S_i)$
$S_{i+1} = E\_K_{gen}(I \text{ xor } K_i^{new})$

Where $E\_K_{gen}(X)$ indicates that the block X is encrypted using the K_{gen} key (the generation key), D is the time stamp and S_i is the seed. K_{gen}, D and S_0 are the initial data.

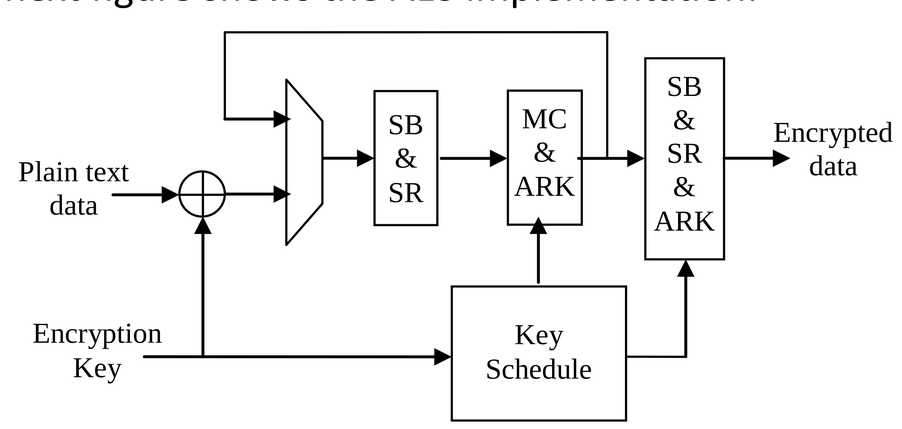
3. Hardware Architecture

The system has been implemented using a Virtex 6 FPGA. The next figure shows the complete system:



The first MicroBlaze takes care of managing the complete GKM and KSM. The second MicroBlaze takes care of calculating new keys (two encryptions are need for one key) and storing in a Key FIFO. MicroBaze 1 take new keys from this Key FIFO.

The next figure shows the AES implementation:



AES has been implementing using pipelining (only for clock reduction purpose), combining phases, parallelizing every byte calculation of a phase and using the *xtime* function to implement the *MixColumn* phase (the most critical one).

4. Results and Conclusions

As we can see in the foloowing tables, we have achieve a low occupation implementation and we have surpassing previous works both in terms of group size and execution time.

	Used
Slices	4,216 (11%)
BlockRAM	86 (20%)
AES Core	
Slices	415 (1.1%)
BlockRAM	16 (3.8%)

Work	Group size	Execution time (ms)
This work	8,388,608	0.028
Shoufan et al.	131,072	3.91
Wong et al.	8,192	1.2
Amir et al.	50	640

5. Future Work

A complete secure architecture will be implemented, which will include MAC, HASH and signing functions. Moreover, to include another MicroBlaze to improve the performance will be studied.

Acknowledgment

This work has been partially funded by the University of Extremadura under contract ACCVII-07.

{granado, mavega , sanperez, jangomez}@unex.es
Department of Technologies of Computers and Communications. University of Extremadura, Cáceres 10003 SPAIN