Performance evaluation of software ciphering in UMTS radio network controller

Master's Thesis, Jukka Jääskeläinen
Nokia Networks

Supervisor: Prof. Timo Korhonen

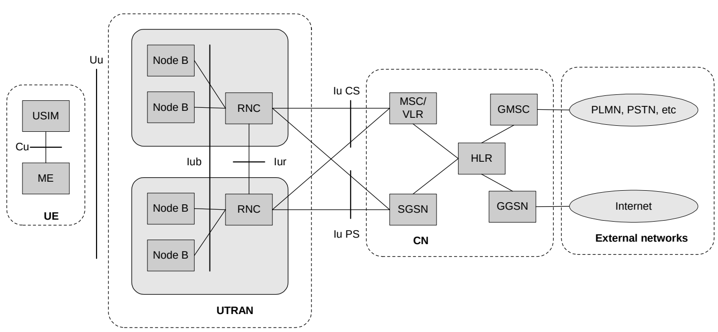
Agenda

- Objectives of the thesis
- Basic UMTS network architecture
- Confidentiality and integrity protection in the UMTS radio access network
- Performance measurement methods
- Results of the study
- Analysis of the results
- Conclusions

Objectives of the thesis

- The purpose of the study is to find out whether the software implementation of UMTS radio access network encryption is feasible
- Feasibility is evaluated primarily from the performance and capacity point of view

UMTS network architecture (Release 99)



- UMTS system is divided into logical entities
  - Core Network (CN)
  - UMTS Terrestrial Radio Access Network (UTRAN)
  - User Equipment (UE)
- External networks are connected to CN via gateway elements

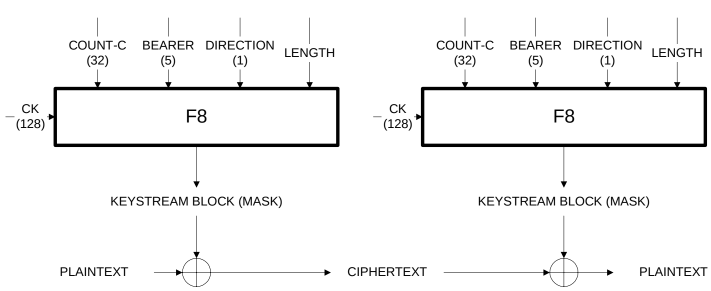
Radio access network encryption and integrity protection

- Cornerstone is the 128-bit secret key ***K***
  - ***K*** is a shared secret between USIM smart card in user's terminal and Authentication Center in user's home network
  - The keys used in encryption and integrity protection are derived from this key
- Data is transferred encrypted between a terminal and a radio network controller (RNC)
  - In GSM the encryption was terminated already in base station (BS) leaving the potentially vulnerable links between BS and Base Station Controller (BSC) unencrypted
- Encryption and integrity protection are symmetric operations, thus exactly the same algorithm is executed both in terminal and in RNC

NOKIA

Confidentiality algorithm – f8

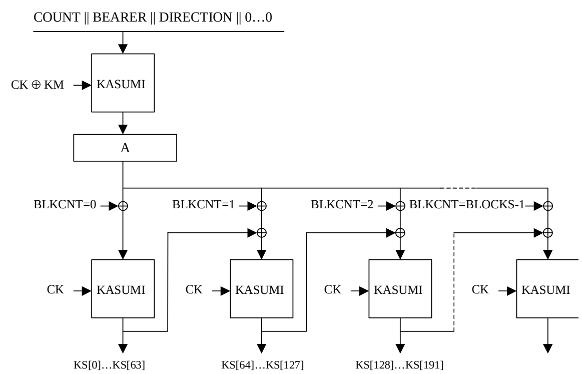
- f8 is a stream cipher being able to encrypt/decrypt blocks of data between 1 and 20000 bits in length
- Algorithm takes five input parameters and generates random-looking mask that is applied to the plaintext
- Internally f8 uses KASUMI block cipher



NOKIA

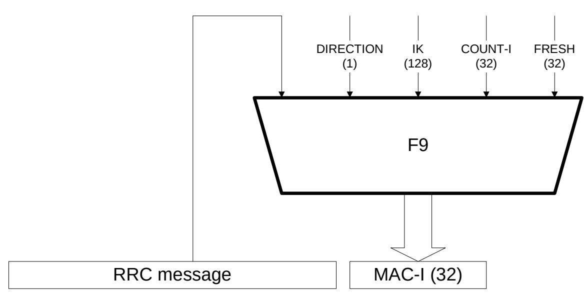
Confidentiality algorithm – f8 (cont.)

- KASUMI block cipher is applied as many times as necessary, one KASUMI round produces 64-bit mask
- As a result keystream (KS) is generated



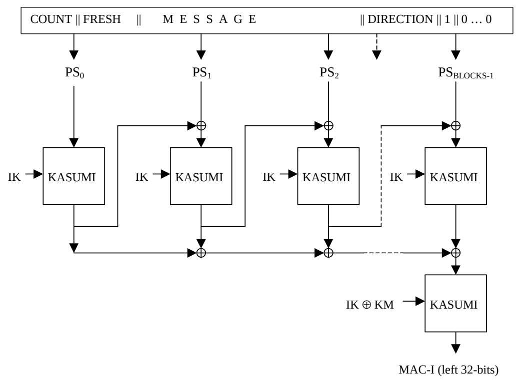
Integrity algorithm – f9

- f9 algorithm is used to implement the integrity protection between a terminal and a network
  - Sending party uses f9 to generate message authentication code (MAC-I)
  - Receiving party uses f9 as well to verify the identity if the sender
- Algorithm takes five input parameters and produces the integrity code that is appended to the end of signaling message



Integrity algorithm – f9 (cont.)

- KASUMI algorithm is also utilized in f9
- The result is 32-bit integrity code MAC-I

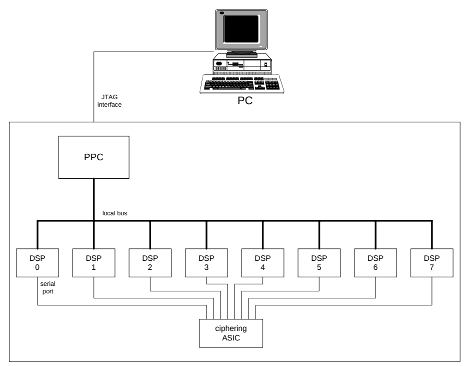


Performance measurements

- A ciphering software module was implemented for the tests
  - Based on the reference implementation in 3GPP TS 35.202
  - Provides full f8 and KASUMI algorithm functionalities
  - Coded in C, not manually optimized
- An existing hardware-based ciphering implementation serves as a reference
  - Ciphering mask generation (i.e. the f8 algorithm) is done in a separate ASIC circuit
- A test process was also implemented
  - Test process uses both the software ciphering and the hardware ciphering and measures the performance
  - Performance is measured in terms of execution time
  - Average, minimum and maximum execution times are measured

Test environment

- A board with eight Texas Instruments TMS320C55X family DSPs
- The ciphering ASIC connected to DSPs via serial interface
  - ASIC driver process is running in each DSP
- A PC connected to the board via JTAG test interface
  - Used for debugging, program loading, result fetching, etc.

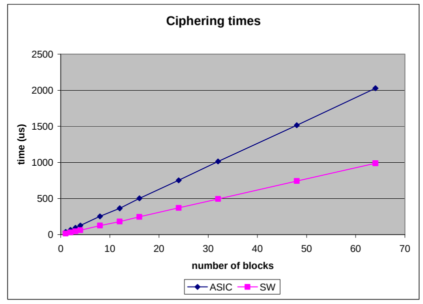


Tests

- Several different kinds of tests were conducted
  - Variable number and size of data blocks to be ciphered
- Most relevant ones map into the data rates and sizes used in real world, i.e. in UMTS
  1. Speech traffic simulation test
     - Data block size is selected to be similar to those used in AMR speech call
  2. Non real-time (NRT) data traffic simulation test
     - Data block sizes are selected to be similar to those in NRT data calls with different data rates

Speech call simulation test

- Measurement results show that software ciphering is significantly faster
- With three data blocks (same in speech call) the software ciphering consumes about half of the time used by ASIC
- Difference behaves linearly being about 50 % throughout the tested range



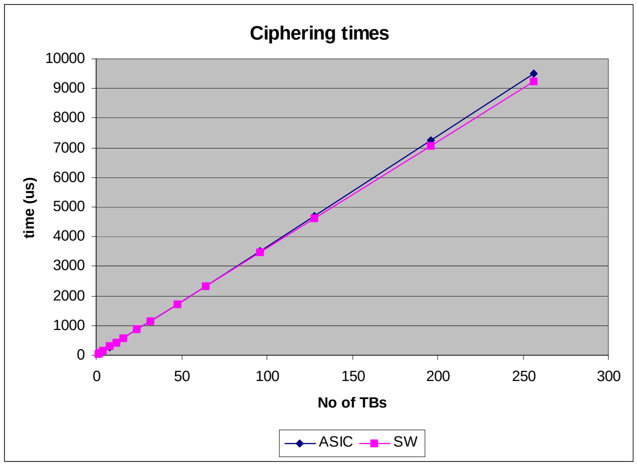
no of blocks	bits / block	ASIC ciph times (us) Min	ave	max	SW ciph times (us) min	ave	max	ave diff (us)	ave diff (%)
1	88	37	37	38	15	15	26	22	59.46
2	88	65	65	66	31	31	41	34	52.31
3	88	95	95	96	46	46	56	49	51.58
4	88	126	126	127	61	61	72	65	51.59
8	88	252	252	253	123	124	133	128	50.79
12	88	360	364	380	180	181	190	183	50.27
16	88	502	502	504	245	247	255	255	50.80
24	88	752	752	756	367	371	378	381	50.66
32	88	1013	1013	1025	490	494	500	519	51.23
48	88	1515	1515	1529	735	742	745	773	51.02
64	88	2027	2027	2042	980	989	990	1038	51.21

NRT data call simulation test

- Measurement results show that the performance is almost the same with both alternatives
- With only a few blocks of more than 50 blocks the software is faster, otherwise the ASIC is marginally faster
- No significant differences

no of blocks	bits / block	ASIC ciph times (us) Min	ave	max	SW ciph times (us) min	ave	max	avg diff (us)	avg diff (%)
1	336	45	46	56	36	36	46	10	21.74
2	336	78	78	90	72	72	82	6	7.69
4	336	142	143	159	143	144	153	-1	-0.70
8	336	284	286	302	286	288	296	-2	-0.70
12	336	425	429	445	429	433	439	-4	-0.93
16	336	568	573	587	572	577	582	-4	-0.70
24	336	853	861	873	858	866	868	-5	-0.58
32	336	1143	1149	1161	1154	1155	1165	-6	-0.52
48	336	1720	1732	1748	1726	1733	1737	-1	-0.06
64	336	2308	2317	2342	2308	2311	2319	6	0.26
96	336	3487	3497	3522	3462	3467	3473	30	0.86
128	336	4678	4690	4715	4616	4622	4627	68	1.45
196	336	7225	7237	7266	7077	7078	7088	159	2.20
256	336	9477	9491	9524	9242	9245	9253	246	2.59

NRT data call simulation test (cont.)



Analysis of the results

- According to the results the software ciphering has at least as good performance than the ASIC ciphering
  - Especially when the number of data frames is small or the data frame size is small
- ASIC solution performance suffers from relatively large overhead in inter-process communication and operating system context switches
  - The ASIC solution involves a lot of signaling between the application process and the ASIC driver process
  - The software ciphering does not have any of this overhead because all the processing is done inside the application process

Pros and cons

ASIC pros:

- Already existing solution, tested and integrated

ASIC cons:

- Lower performance due to the interface overhead

SW pros:

- No need for HW design
- Better performance
- Flexible, new functionality can be added later if needed
  - New algorithms etc.
- Rather straightforward to test

SW cons:

- Consumes some of the DSP processing power (max ~8 %)

Conclusions

- Software ciphering improves performance, especially for speech traffic ciphering
- It also simplifies the architecture
  - No need for HW-SW interface
  - Faster design cycle
- Implementation is found to be straightforward and require a reasonable amount of time

Thus, the software ciphering is estimated to be a very feasible choice for the purpose.

