On Hardware-Assisted Cryptanalysis of A5/1

Estonian Computer Science Theory Days

Viinistu, Oct 28 2005

Emilia Käsper

Helsinki University of Technology

Agenda

- A5/1: introduction/reminder
- A Guess-and-determine attack on A5/1
- Complexity analysis of the attack
- Implementations

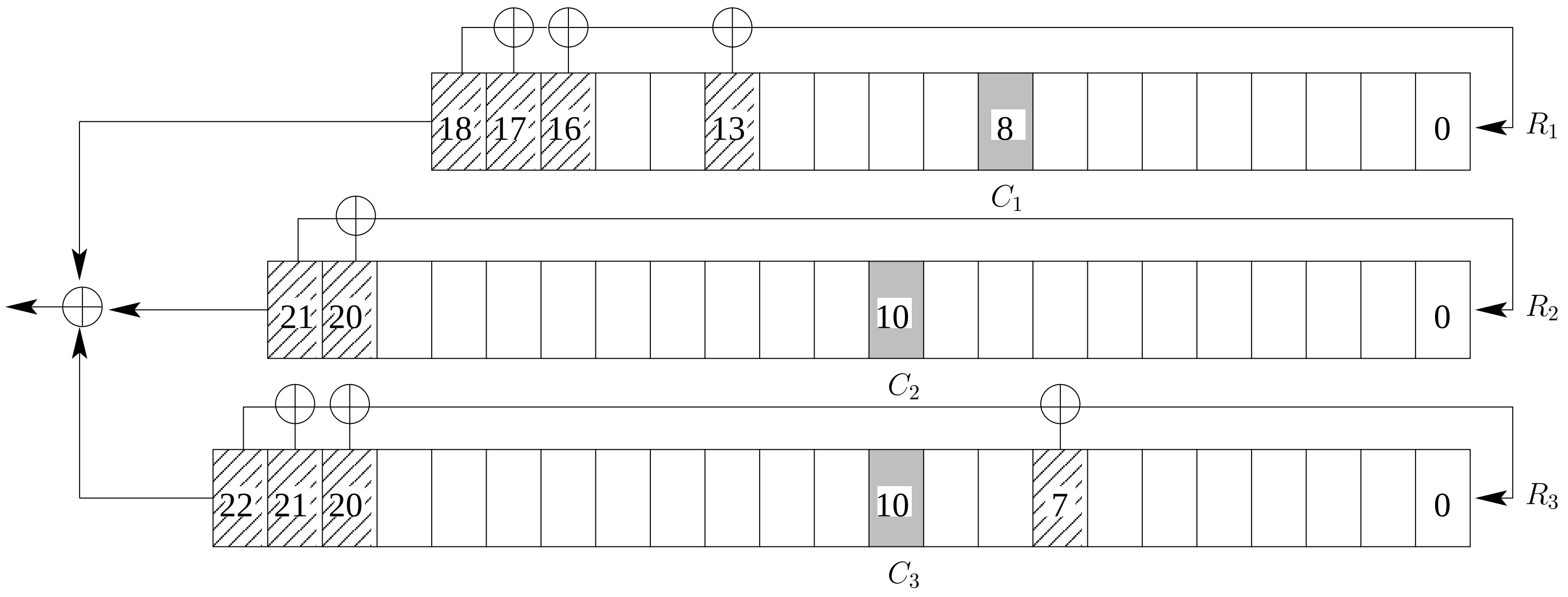
The A5 Family of Stream Ciphers

- **A5/0**: Dummy cipher, no encryption
- **A5/1**: Most widespread currently and in the near future
- **A5/2**: Weaker version designed for export, in removal
- **A5/3**: 3G algorithm

A5 Encryption

- **Input**: 64-bit secret key K_c, 22-bit publicly known IV F_n
- **Output**: Two 114-bit blocks of keystream used to encrypt uplink and downlink data
- **Data**: Signalling information or 5 ms of digitally encoded speech
- **Encryption**: Bitwise exclusive-or

The A5/1 Encryption Algorithm



A5/1: Key Setup

1. Clock the three registers regularly for 64 cycles; xor key bits into the least significant bits (lsb-s) of the registers
2. Clock regularly for another 22 cycles; xor IV bits into lsb-s of the registers $\rightarrow$ **initial state**
   - The initial state is a linear combination of key and IV bits
   - Knowledge of the initial state reveals the key

A5/1: Cipher Operation

3. Clock irregularly for 100 cycles; discard output $\rightarrow$ **internal state** at time $t = 0$
   - Irregular majority-based clocking rule:
     clock register R_i if $C_i = \text{majority}(C_1, C_2, C_3)$
4. Clock irregularly for 228 cycles; output 228 keystream bits

Attacking A5/1

1. Recover the internal state of the cipher at time $t = k$
2. If $k \neq 0$, invert the algorithm to recover the possible internal state(s) at time $t = 0$
3. Invert key setup to recover the possible initial state(s) of the cipher $\rightarrow$ key K_c revealed
4. Verify result(s) with a different IV F_n'

- ▶ Steps 2.-4. are easy compared to Step 1.

Guess-and-Determine Attacks

- Internal state recovery attack
- Known plaintext model (NB! Known plaintext = known keystream)
- Guess some of the 64 bits of the internal state at some time $t = k$
- Recover the remaining bits from known keystream

Anderson-Keller-Seitz Attack

- Guess the 41 bits of registers R_1 and R_2
- Determine the 23 bits of register R_3 from known keystream
- Check result against further keystream
- But—the 23 bits are not uniquely determined (yet)
- **Question**: how many additional bits do we need to guess?

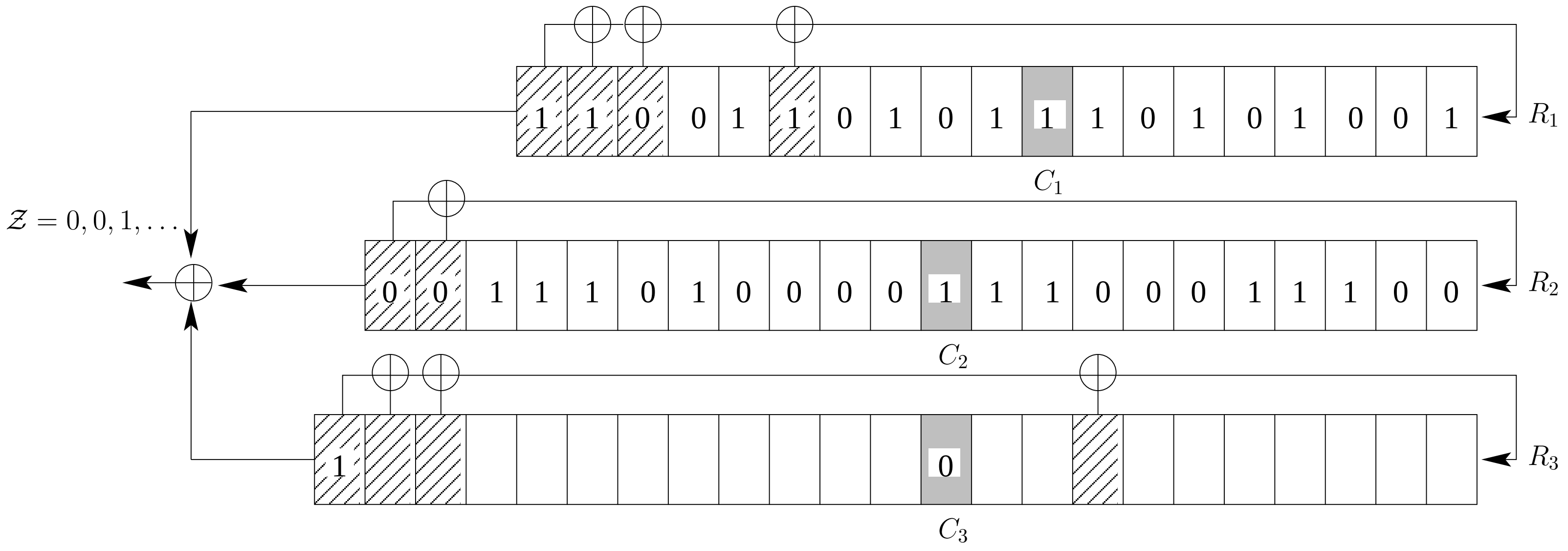
AKS Attack II

- The 41 bits of registers R_1 and R_2 plus the 11 less significant bits of R_3 uniquely determine the clocking for 11 cycles
- Now, known keystream uniquely determines the remaining 12 bits
- This gives a complexity of 2^{52} bit guesses
- Can we do better?

AKS Attack III: Complexity Analysis

- Start guessing the 11 bits of R_3 one by one
- We proved that with probability $\frac{2}{7}$, the next bit of R_3 is uniquely determined
- Therefore, we need on average $\left(\frac{2}{7} \cdot 1 + \frac{5}{7} \cdot 2\right)^{11} \approx 2^{8.6}$ bit guesses
- The overall average-case complexity of the attack is thus $\frac{1}{2} \cdot 2^{41+8.6} = 2^{48.6}$ bit guesses

AKS Attack IV: Example



Comparison of Implementations

	AKS Software	**AKS Hardware**
Author	Käsper and Lipmaa	Keller and Seitz
Year	2005	2001
Platform	Intel Celeron M 1.3GHz	Xilinx XC4062 FPGA
Attack time	2.5 years	2 years (our estimate)

Conclusions

- The complexity of the AKS guess-and-determine attack is $2^{48.6}$ bit guesses
- Software and hardware implementations comparable in performance (no pipelining effect in hardware)
- However, PCs get only faster, but FPGA-s get faster *and* larger (parallelization effect)

Thank You! Questions?