Stream ciphers — what does industry want?

Steve Babbage
Vodafone Group R&D
steve.babbage@vodafone.com

vodafone™

Overview

- Why not just use AES?
- Characteristics of a stream cipher
- Where stream ciphers can provide real benefit
- Other requirements

Why not just use AES?

- There are clear requirements for stream cipher systems
  - e.g. radio communications where zero error propagation is highly desirable
- But AES in Counter Mode or Output Feedback Mode gives us a nice, standard, pretty fast, pretty efficient stream cipher

- Some industry sector representatives saw no particular requirements from their sectors for dedicated stream ciphers

- Are stream ciphers just an interesting but unnecessary alternative to AES for certain applications?
- Or can they meet genuine requirements that AES cannot?

Overview

- Why not just use AES?
- Characteristics of a stream cipher
- Where stream ciphers can provide real benefit
- Other requirements

Stream cipher functionality: the basic parameters

- Secret key size
- IV size
  - Almost every real application of a stream cipher requires a single secret key to be used many times, with a different IV
  - May be called “nonce” or “message key”
- Maximum output length
  - How long a keystream sequence does it need to produce?

Stream cipher functionality: performance

Speed …	… and size …	… on different platforms
Initialisation time		32-bit or 64-bit processor
Re-initialisation time (same secret key, new IV)	Implementation size	8-bit processor
	Power consumption	FPGA
Throughput		Purpose built ASIC

Stream cipher functionality: special

- Transform plaintext to ciphertext by bitwise XOR? Or a more complex transformation on larger plaintext/ciphertext blocks?
- Direct keystream access? (Generate the millionth keystream block efficiently without cycling through a million generator states)
- Other functionality combined with the stream cipher, e.g. an integrity mechanism?

Overview

- Why not just use AES?
- Characteristics of a stream cipher
- Where stream ciphers can provide real benefit
- Other requirements

Extreme performance

- AES in counter or OFB mode can indeed meet most requirements
- Where dedicated stream ciphers may do better is at the performance extremes:
    - Very high speed: multi-Gigabit-per-second communications links (e.g. routers)
        - May be all in software …
        - … or may use hardware accelerators for entire algorithm or for some functional components
        - Ability to parallelise is good for speed, but more hardware = higher cost
    - Efficient / compact in constrained devices:
        - Very small battery powered devices e.g. RFID
        - Smart cards (8-bit processors)
    - Different observers rated the importance of high speed / small hardware size differently

Authenticated encryption

- Big requirement for efficient authenticated encryption
- One contributor observed:
  - Today AES can run in 🕓20 clocks per byte (cpb) on a Pentium ...
  - ... but HMAC-SHA takes 14-15 cpb …
  - … so even if we speed up AES by a factor of four, we don't even double the combined speed
- An efficient combined encryption and authentication algorithm would have real value …
  - … whether it’s a “stream cipher plus” …
  - … or a block cipher mode
    - N.B. Several block cipher modes for authenticated encryption have been proposed, but the most efficient (OCB and its predecessors) are heavily encumbered with IPR
  - Again parallelisable is good

Overview

- Why not just use AES?
- Characteristics of a stream cipher
- Where stream ciphers can provide real benefit
- Other requirements

Stream cipher functionality: the basic parameters

- Secret key size

Generally 128 bits — although this gives a lower limit on hardware size

Support large IVs, but work well (perhaps more efficiently) with smaller ones

- IV size
  - Almost every real application of a stream cipher requires a single secret key to be used many times, with a different IV
  - May be called "nonce" or "message key"

AES in Counter or OFB mode shows trivial distinguishability from random after 2^{64} blocks
2^{64} is more than enough before reinitialising with a new IV

- Maximum output length
  - How long a keystream sequence does it need to produce?

vodafone™

Stream cipher functionality: performance

Speed …	… and size …	… on different platforms
Initialisation time		32-bit or 64-bit processor
Re-initialisation time (same secret key, new IV)	Implementation size	8-bit processor
	Power consumption	FPGA
Throughput		Purpose built ASIC

Both very important — don't just look at throughput

Stream cipher functionality: special

No requirement for anything other than bitwise XOR

- Transform plaintext to ciphertext by bitwise XOR? Or a more complex transformation on larger plaintext/ciphertext blocks?

Slightly useful, but not very. (Stream ciphers not suitable for databases — "Whoever heard of a write-once database?")

- Direct keystream access? (Generate the millionth keystream block efficiently without cycling through a million generator states)
- Other functionality combined with the stream cipher, e.g. an integrity mechanism?

Acknowledgements and conclusions

- Particular thanks to Doug Whiting and Greg Rose
- The feedback received from industry closely matched internal assessments performed within the ECRYPT consortium
- All this is reflected in the “way forward” proposals that you will see later in this conference